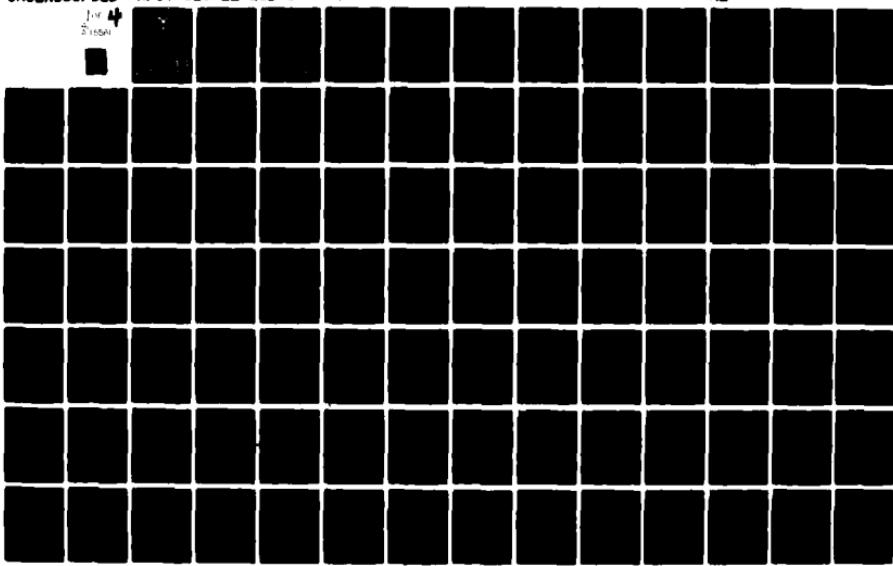


AD-A115 581 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/B 12/1
ENHANCED TRACKING OF AIRBORNE TARGETS USING FORWARD LOOKING INF--ETC(U)
DEC 81 S K ROGERS
AFIT/SE0/EE/81D-5

UNCLASSIFIED

NL

for 4
Sloan

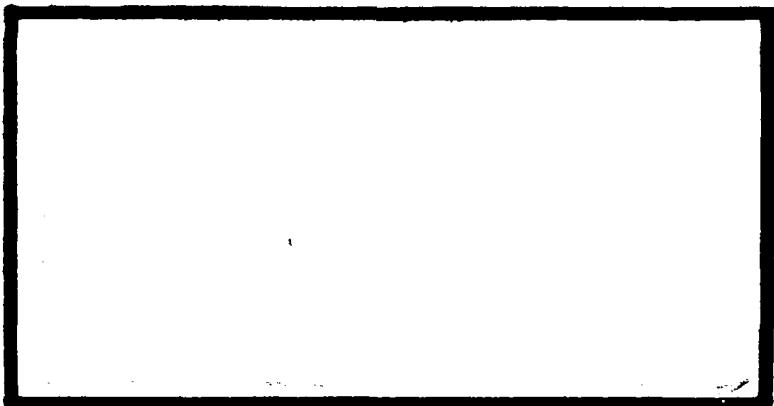


DTIC FILE COPY

- ADA115581



PDC
①



DTIC
ELECTED
S JUN 15 1982 D

E

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY (ATC)

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

This document has been approved
for public release and sale; its
distribution is unlimited.

82 06 14 174

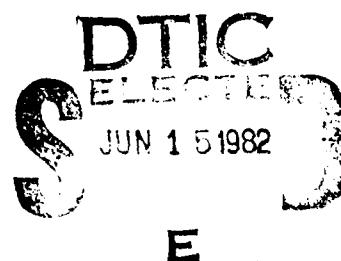
AFIT/GEO/EE/81D-5

(1)

ENHANCED TRACKING OF AIRBORNE
TARGETS USING FORWARD
LOOKING INFRARED MEASUREMENTS

THESIS

AFIT/GEO/EE/81D-5 Steven K. Rogers
 1st Lt USAF



ENHANCED TRACKING OF AIRBORNE
TARGETS USING FORWARD
LOOKING INFRARED MEASUREMENTS

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Electrical Engineering

by

Steven K. Rogers B.S.E.E.

1st Lt USAF

Graduate Electro-Optics

December 1981

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/ _____	
Availability Codes	
Dist	Avail and/or Special
A	



Preface

This study was part of a continuing effort to design next generation trackers for one of the Air Force Weapons Laboratory's laser weapons using modern optimal estimation and control techniques. This report uses these techniques to develop an extended Kalman filter which uses outputs of a forward looking infrared sensor in a tracking problem. As an alternate tracker, an extended Kalman filter is designed to process the position estimate outputs of a correlation algorithm.

I wish to express my thanks to Dr. Peter S. Maybeck, my advisor, for his motivation, expert advice and valuable time. Mr. David R. McGrew deserves additional thanks. His knowledge of fundamental engineering concepts proved invaluable to the development and testing of the data processing algorithms used in this research. I wish also to thank James Emett "Microman" Crider and Robert Suizu for their assistance in using the Modcomp minicomputer.

The author reserves a very special thanks to his father who continually gives him an unlimited source of inspiration. This thanks is also extended to his mother as the proverbial source of tender loving care. To his wife, whose support was absolutely essential for the completion of all his studies from the early days of midnight factory shifts and daytime classes to today, the author has only his never ending and

passionate love. To my beautiful baby daughter, Sarah, it certainly is nice to always have someone to come home to who will give me a pretty dimpled smile. Last but definitely most, to his primary source of pride and everlasting joy, his son, Jeremiah, who always had to wait for his Dad to come home or finish studying before they could go out to play together, the author has his infinite love and sincerest apologies. I would also like to acknowledge my typist, Ms Cheryl Nicol. Her professionalism and dedication made completion of this document in time to meet an early deadline possible and was greatly appreciated.

Steven K. Rogers

Contents

	Page
Preface	ii
List of Figures	vii
List of Tables	xii
List of Symbols	xiii
Abstract	xviii
I. Introduction	1
Background	2
Problem	4
Plan of Attack	9
Overview	12
II. Derivation of the Nonlinear and Linearized Intensity Functions	14
Introduction	14
Pattern Recognition	14
Two-Dimensional Fourier Transforms	16
Shifting Property of Fourier Transforms	18
Smoothing in the Fourier Domain	20
Derivative Property of Fourier Transforms	21
Derivation of Intensity Functions	23
III. Truth Model	25
Introduction	25
Target Dynamics Model	25
Atmospheric Jitter Model	27
State Space Model	29
Propagation Equations	32
Measurements	34
Spatially Correlated Background Noise	36
Summary of Truth Model	40
IV. Extended Kalman Filter	41
Introduction	41
Target Dynamics and Atmospheric Jitter Filter Models	41
State Space Model	42

Contents

	Page
State Propagation	44
Measurement Update Equations	47
Summary of the Extended Kalman Filter Equations	50
V. Correlator-Kalman Filter Tracker	52
Introduction	52
Correlator Implementation	53
Correlator Error Statistics	63
Kalman Filter Tracker	68
VI. Performance Analysis	71
Introduction	71
Derivation of Intensity Functions	72
Tracking Ability	75
Variation of Parameters	77
Plotting Results	81
Table Summary	99
Summary	112
VII. Optical Processing Alternatives	113
Introduction	113
Background	114
Applying Optical Processing to Tracking	118
Conclusion	123
VIII. Conclusions and Recommendations	125
Conclusions	125
Recommendations	126
Bibliography	129
Appendix A: Two-Dimensional Finite Discrete Fourier Transform Interpretation and Test	131
Appendix B: Shifting Property of the Two-Dimensional Finite Discrete Fourier Transform	146
Appendix C: Implementation of the Exponential Smoothing Algorithm	151

Contents

	Page
Appendix D: Implementation of the Spatial Derivatives Using Fourier Transforms	155
Appendix E: Generation of White Gaussian Noise Process	161
Appendix F: Discrete Representation of the Derivative of the Intensity Profile with Respect to the Kalman Filter States	164
Appendix G: Monte Carlo Study	169
Appendix H: Computer Software	171
Appendix I: Plots of Tracking Errors	269
Appendix J: Kalman Filters	334

List of Figures

<u>Figure</u>		<u>Page</u>
1	Data Processing Algorithm	6
2	Research Plan of Attack Flow Diagram	10
3	Third Order Shaping Filter	28
4	FLIR Data Frame Pixel Numbering Scheme	37
5	First and Second Nearest Neighbor	38
6	Centered Single Gaussian Template	56
7	Noise Corrupted Data Array	57
8	Result of Cross Correlation of Data and Template	58
9	Diagonal Quadrant Swap of Cross Correlation	60
10	Result of Thresholding of .5	62
11a	Histogram of Error in Horizontal Position Estimate	65
11b	Histogram of Error in Vertical Position Estimate	66
12	X Minus Errors	84
13	Y Minus Errors	85
14	X Plus Errors	86
15	Y Plus Errors	87
16	X Position Error	88
17	Y Position Error	89
18	X Centroid Minus Error	90
19	Y Centroid Minus Error	91
20	X Centroid Plus Error	92

List of Figures

<u>Figure</u>		<u>Page</u>
21	Y Centroid Plus Error	93
22	X Centroid Position Error	94
23	Y Centroid Position Error	95
24	Error of Estimated H	96
25	Error of Estimated DH/DX	97
26	Error of Estimated DH/DY	98
27	Basic Optical Processing System	115
E-1	Output Probability Density of Pseudorandom Code	161
F-1a	Continuous One-Dimensional Gaussian	166
F-1b	Discrete One-Dimensional Gaussian	166
F-2	Shifted Intensity Profile	167
F-3	Intensity's Spatial Derivative	167
F-4	Derivative with Respect to the States	168
G-1	Generation of Error Samples	169
I-1	Pad Zeros X Minus Errors	271
I-2	Pad Zeros Y Minus Errors	272
I-3	Pad Zeros X Plus Errors	273
I-4	Pad Zeros Y Plus Errors	274
I-5	Pad Zeros X Position Errors	275
I-6	Pad Zeros Y Position Errors	276
I-7	Pad Zeros X Centroid Minus Errors	277
I-8	Pad Zeros Y Centroid Minus Errors	278
I-9	Pad Zeros X Centroid Plus Errors	279

List of Figures

<u>Figure</u>		<u>Page</u>
I-10	Pad Zeros Y Centroid Plus Errors	280
I-11	Pad Zeros X Centroid Position Errors	281
I-12	Pad Zeros Y Centroid Position Errors	282
I-13	Pad Zeros Error of Estimated h	283
I-14	Pad Zeros Error of Estimated Dh/Dx	284
I-15	Pad Zeros Error of Estimated Dh/Dy	285
I-16	Removing Two Highest Spatial Frequencies X Minus Errors	287
I-17	Removing Two Highest Spatial Frequencies Y Minus Errors	288
I-18	Removing Two Highest Spatial Frequencies X Plus Errors	289
I-19	Removing Two Highest Spatial Frequencies Y Plus Errors	290
I-20	Removing Two Highest Spatial Frequencies X Position Errors	291
I-21	Removing Two Highest Spatial Frequencies Y Position Errors	292
I-22	Removing Two Highest Spatial Frequencies X Centroid Minus Errors	293
I-23	Removing Two Highest Spatial Frequencies Y Centroid Minus Errors	294
I-24	Removing Two Highest Spatial Frequencies X Centroid Plus Errors	295
I-25	Removing Two Highest Spatial Frequencies Y Centroid Plus Errors	296
I-26	Removing Two Highest Spatial Frequencies X Centroid Position Errors	297
I-27	Removing Two Highest Spatial Frequencies Y Centroid Position Errors	298

List of Figures

<u>Figure</u>		<u>Page</u>
I-28	Removing Two Highest Spatial Frequencies Error of Estimated h	299
I-29	Removing Two Highest Spatial Frequencies Error of Estimated Dh/Dx	300
I-30	Removing Two Highest Spatial Frequencies Error of Estimated Dh/Dy	301
I-31	Removing Four Highest Spatial Frequencies X Minus Errors	303
I-32	Removing Four Highest Spatial Frequencies Y Minus Errors	304
I-33	Removing Four Highest Spatial Frequencies X Plus Errors	305
I-34	Removing Four Highest Spatial Frequencies Y Plus Errors	306
I-35	Removing Four Highest Spatial Frequencies X Position Errors	307
I-36	Removing Four Highest Spatial Frequencies Y Position Errors	308
I-37	Removing Four Highest Spatial Frequencies X Centroid Minus Errors	309
I-38	Removing Four Highest Spatial Frequencies Y Centroid Minus Errors	310
I-39	Removing Four Highest Spatial Frequencies X Centroid Plus Errors	311
I-40	Removing Four Highest Spatial Frequencies Y Centroid Plus Errors	312
I-41	Removing Four Highest Spatial Frequencies X Centroid Position Errors	313
I-42	Removing Four Highest Spatial Frequencies Y Centroid Position Errors	314
I-43	Removing Four Highest Spatial Frequencies Error of Estimated h	315

List of Figures

<u>Figure</u>		<u>Page</u>
I-44	Removing Four Highest Spatial Frequencies Error of Estimated Dh/Dx	316
I-45	Removing Four Highest Spatial Frequencies Error of Estimated Dh/Dy	317
I-46	Maximum Noise X Minus Errors	319
I-47	Maximum Noise Y Minus Errors	320
I-48	Maximum Noise X Plus Errors	321
I-49	Maximum Noise Y Plus Errors	322
I-50	Maximum Noise X Position Errors	323
I-51	Maximum Noise Y Position Errors	324
I-52	Maximum Noise X Centroid Minus Errors	325
I-53	Maximum Noise Y Centroid Minus Errors	326
I-54	Maximum Noise X Centroid Plus Errors	327
I-55	Maximum Noise Y Centroid Plus Errors	328
I-56	Maximum Noise X Centroid Position Errors	329
I-57	Maximum Noise Y Centroid Position Errors	330
I-58	Maximum Noise Error of Estimated h	331
I-59	Maximum Noise Error of Estimated Dh/Dx	332
I-60	Maximum Noise Error of Estimated Dh/Dy	333

List of Tables

<u>Table</u>		<u>Page</u>
1	Modcomp Tracking and Intensity Function Derivation Errors	100
2	Cyber Tracking and Intensity Function Derivation Errors	102
3	Modcomp Tracking Errors for Correlator- Kalman Filter Algorithm	105

List of Symbols

Symbol

$\hat{x}(t_i^+)$	state estimate vector after measurement incorporation
$\hat{x}(t_i^-)$	state estimate vector before measurement incorporation
$K(t_i)$	Kalman filter gain
$z(t_i)$	measurement vector of average intensities over individual pixel elements (pixels) of the FLIR
$h(\hat{x}(t_i), t_i)$	intensity shape function for measurements at time t_i as a function of the state estimate
x_{centroid}	x-position of the centroid of the target's intensity profile
x_{dynamics}	x-position of the centroid resulting from target dynamics
$x_{\text{atmospherics}}$	x-position of the centroid resulting from atmospheric jitter
$\tilde{G}(f_x, f_y)$	Frequency Spectrum of two-dimensional sequence $\tilde{g}(x, y)$
$\tilde{g}(x, y)$	spatial domain representation
f_x, f_y	spatial frequencies
x, y	space variables
$\hat{y}(t)$	current averaged value
$y(t)$	current data frame
$\hat{y}(t-1)$	previous averaged data frame
α	smoothing constant
T_t	target correlation time

List of Symbols

<u>Symbol</u>	
$\omega_1(t), \omega_2(t)$	continuous time, independent, white Gaussian noise processes
σ_d^2	desired variance on outputs x_d and y_d
F_t	truth model plant matrix
$\underline{x}_t(t)$	truth model state vector
G_t	truth model noise input matrix
w_t	vector of white Gaussian noise inputs
Φ_t	truth model state transition matrix
$z_{K\ell}$	output of the $K-\ell$ -th pixel
A_p	area of one pixel
I_{\max}	maximum intensity received from target
σ_g^2	dispersion of the Gaussian intensity function
$x_{\text{peak}}(t_i), y_{\text{peak}}(t_i)$	center of the Gaussian intensity pattern
$v_{K\ell}(t_i)$	additive noise for the $K-\ell$ -th pixel corresponding to background and FLIR noises
d_{ij}	distance in # pixels from pixel i to pixel j
$\underline{v}'(t_i)$	white Gaussian vector each component zero mean, unit variance, and independent of other components
T_f	correlation time
$\omega_f(t)$	white Gaussian noise process
σ_f	root mean squared value of x
F_f	filter plant matrix
$\underline{x}_f(t)$	filter state vector

List of Symbols

Symbol

$\underline{w}_f(t)$	input white Gaussian noise vector for filter
T_{df}	correlation time assumed for target dynamics
T_{af}	correlation time assumed for atmospheric jitter
σ_{df}^2	assumed target dynamics noise variance
σ_{af}^2	assumed atmospheric jitter noise variance
Φ_f	filter state transition matrix
$P(t_i^+)$	conditional state covariance matrix from measurement update equation at time t_i
$P(t_{i+1}^-)$	conditional state covariance matrix propagated from time t_i to t_{i+1}
Ω_f	noise covariance Kernel descriptor given in Equation (4-8)
$\underline{H}(t_i)$	$\partial \underline{h}(\hat{x}(t_i), t_i) / \partial \underline{x}$ linearized function of intensity measurements
$K(x, y)$	output of cross correlation
C	centroid summation
\underline{z}_c	estimated x and y coordinates by the correlation/centroid algorithm
\underline{h}_c	the linear combination of state variables which contribute to the respective measurement elements
\underline{x}_c	correlator state vector
\underline{v}_c	additive noise corruption to correlation position estimates
M_{eij}	mean error for pixel j at the time frame i
$\underline{\mathcal{C}}_R$	Cholesky square root

List of Symbols

Symbol

$\underline{P}(t_i^-)$	propagated conditional covariance matrix before measurement incorporation at time t_i
$\hat{\underline{x}}(t_i^-)$	propagated state estimate vector of filter before measurement incorporation at time t_i
$K(t_i)$	Kalman filter gain
$h(\hat{\underline{x}}(t_i^-), t_i)$	nonlinear function of intensity measurements at time t_i , as function of filter state estimates $\hat{\underline{x}}(t_i^-)$
$\underline{z}(t_i)$	actual realization of measurement vector
$g(x,y) * \underline{l}(x,y)$	cross correlation of the two-dimensional spatial sequences $g(x,y)$ and $\underline{l}(x,y)$
$\underline{L}^*(f_x, f_y)$	complex conjugate of the Fourier transform of the sequence $\underline{l}(x,y)$
$\underline{z}_c(t_i)$	the estimated x and y coordinates by the correlation/centroid algorithm
\underline{H}_c	the linear combination of state variables which contribute to the respective measurement elements
\underline{x}_c	the state vector
$\underline{v}_c(t_i)$	additive noise corruption assumed to be white and Gaussian with statistics to be determined
N	number of simulations
k	index of Monte Carlo simulation runs going from 1 to N
h_{tijk}	value for frame i, pixel j, of the truth model intensity for the kth simulation
\hat{h}_{tijk}	estimate value for frame i, pixel j, of the intensity as derived in the kth simulation
M_{ei}	the spatial average (over all the pixels) of the mean pixel errors at the ith frame

List of Symbols

Symbol

$\sigma_{e_i}^2$	the spatial average of all the ensemble pixel error variances, over all the pixels at the ith frame
$\sigma_{e_{ij}}^2$	variance of the error of jth pixel at the ith frame
$\bar{E}_{x_d_i}$	mean error in x dynamics for frame i
$x_{t_{d_{ik}}}$	truth model, x dynamic value at frame i simulation k
$\hat{x}_{d_{ik}}$	filter estimated x dynamic value at frame i simulation k
K_1	complex constant
T_0	Fourier transform of $t_0(x_1, y_1)$
τ	mean wavelength of the laser
α, K_0	constants
$\underline{L}(t_i)$	current averaged data frame in frequency domain
$\underline{L}(t_i)$	transform of current noise corruption data frame
$\hat{\underline{L}}(t_{i-1})$	previous result of averaging in frequency domain

(4) *Adaptive Extended Kalman Filter*

Abstract

Considerable work has been accomplished at AFIT in the last three years to improve the tracking capability of the high energy laser weapon. The improvements were achieved via use of an adaptive extended Kalman filter algorithm. In this research, work is initiated on a tracker able to handle "multiple hot spot" targets, in which digital signal processing is employed on the FLIR data to identify the underlying target shape. This identified shape is then used in the measurement model portion of the filter as it estimates target offset from the center of the field-of-view. Two tracking algorithms are developed. The first algorithm uses an extended Kalman filter to process the intensity measurements from a FLIR to produce target position estimates. The second algorithm uses a linear Kalman filter to process the position estimates of an improved correlation algorithm. This algorithm is improved over standard correlators by using thresholding to eliminate poor correlation information, dynamic information from the Kalman filter and it also uses the on-line derived target shape.

ENHANCED TRACKING OF AIRBORNE TARGETS USING FORWARD LOOKING INFRA-RED MEASUREMENTS

I. Introduction

The first successful laser is generally credited to Maiman at the Hughes Research Laboratories in 1960. Since that time, there has been an explosive growth in laser technology. High energy lasers have become prime candidates for use as weapon systems (1:87), and in fact have already been tested against airborne targets (2:14-17; 3:16-19). The use of high energy lasers for military applications may have a significant impact on any future war (1:87).

Laser weapon systems are highly desirable because of their potential to deposit large amounts of energy on a small area of a target in a short period of time. However, there are many technological problems which must be resolved prior to high energy lasers being used as efficient and effective weapon systems.

The problem of very accurate target position estimation, for tracking the target, is one such technical problem. An Infra-Red (IR) laser beam must be kept positioned on a specific part of a target for an extended amount of time to destroy a target or critical components of a target. An aiming angle correct to within "six one hundred thousandths of one degree", .1 microrad, "is required to hit a missile 5000 Km away, which

is a precision beyond existing technology for both the U.S. and the U.S.S.R." (1:87-89).

Background

This research investigates potential solutions to the target tracking problems associated with directed energy beam weapons. The Air Force Weapons Laboratory (AFWL) located at Kirtland Air Force Base, New Mexico, currently uses correlation trackers to provide precise target position estimates to feedback controllers in the presence of several disturbances. These disturbances include any effect that can cause relative motion between the beam and the target, such as target motion, mirror vibration, atmospheric jitter, and sensor measurement errors (4:2). A correlation tracker stores a complete set of predetermined or previous real-time data which provide a template to compare with new information to be received at a later time. Cross correlation techniques are used to estimate the relative position offsets from one frame of data to the next. This relative position information is then used to control the gimbals which keep the target centered within the sensor's field-of-view. One possible sensor, the one currently of most interest, is the Forward Looking Infra-Red sensor (FLIR).

The high energy laser is servoed to the FLIR so that centering the target within the FLIR's field-of-view physically points the laser toward the target. This type of

tracker needs no prior information about the target's shape or dynamics and is therefore well suited to many general applications. The disadvantages of this tracker are its susceptibility to noise and its insensitivity to any knowledge of the target's dynamics.

To overcome these specific deficiencies of the correlator tracker, the use of an extended Kalman filter tracker has been investigated (4;5;6). Appendix J contains a very brief generic view of the Kalman filter equations and then provides information on how they are used for this application. In many practical tracking problems, the type of target being tracked is known along with target parameters such as size, shape and even acceleration characteristics. Additional information incorporated by the Kalman filter, which is not utilized by correlation trackers, is knowledge of the statistical effects of atmospheric distortion on the radiated wave-front as it propagates to the FLIR. The Kalman filter uses this information to aid in separating the true target relative motion from the atmospheric disturbance. The Kalman filter operating upon the raw digitized image has been shown to perform well in comparison to correlator trackers when the target intensity function shape is relatively well known and unimodal, and when the internal filter models depict the actual tracking situation reasonably well (5;6). These studies found the extended Kalman filter tracker to be a

robust tracker, even in low signal-to-noise-ratio environments.

Problem

A major difficulty in using an extended Kalman filter for enhanced IR tracking is the Kalman filter algorithm's requirement for an accurate reference image, $\underline{h}[\underline{x}(t_i), t_i]$ and the derivatives, $\partial[\underline{h}(\underline{x}(t_i), t_i)]/\partial\underline{x} = \underline{H}(\underline{x}(t_i), t_i)$ of that reference image with respect to the states, over the entire FLIR image. This research adopts the notation that an underlined lower case letter denotes a vector while an uppercase letter underlined denotes a matrix. In previous research efforts, this reference image was assumed to be a known Gaussian functional form. During the past year, work was initiated on a tracker able to handle "multiple hot spot" targets, in which digital processing must be employed to identify the target's shape (4:). This identified shape could then be used in the measurement model portion of the Kalman filter as it estimates target offsets from the center of the field-of-view. The state estimates, in turn, are incorporated into the shape function determination. The data processing algorithm is shown in Figure 1. The Kalman filter processes the measurement vector $\underline{z}(t_i)$ using the equation

$$\hat{\underline{x}}(t_i^+) = \hat{\underline{x}}(t_i^-) + \underline{K}(t_i)[\underline{z}(t_i) - \underline{h}(\hat{\underline{x}}(t_i^-), t_i)] \quad (1-1)$$

where

$\hat{x}(t_i^+)$ = state estimate vector after measurement incorporation at time t_i

$\hat{x}(t_i^-)$ = state estimate vector propagated from previous measurement update to time t_i

$K(t_i)$ = Kalman filter gain

$z(t_i)$ = Measurement vector of average intensities over individual picture elements (pixels) of the FLIR array

$h(\hat{x}(t_i^-), t_i)$ = intensity shape function for measurements at the time t_i as a function of the state estimate

The apparent location of the target within the sensor's field-of-view is the sum of effects due to true target dynamics and atmospheric disturbances. The four-state estimate vector, $\hat{x}(t_i)$, consists of estimates for the x and y positions due to true target dynamics and the x and y positions due to atmospherics.

Figure 1 shows two data processing paths for the intensity measurements. The FLIR is positioned so that the center of its field-of-view is where the Kalman filter predicted the true target position to be, resulting from dynamics over the most recent sample period. Each FLIR frame is arranged by rows into a measurement vector $z(t_i)$ which is the input to the Kalman filter in the lower path of Figure 1. The Kalman filter uses the nonlinear intensity function evaluated at the predicted state, $h[\hat{x}(t_i^-), t_i]$, and the linearized intensity function, $H[\hat{x}(t_i^-), t_i]$, for that frame to compute an updated state estimate, $\hat{x}(t_i^+)$, from the measurement vector $z(t_i)$.

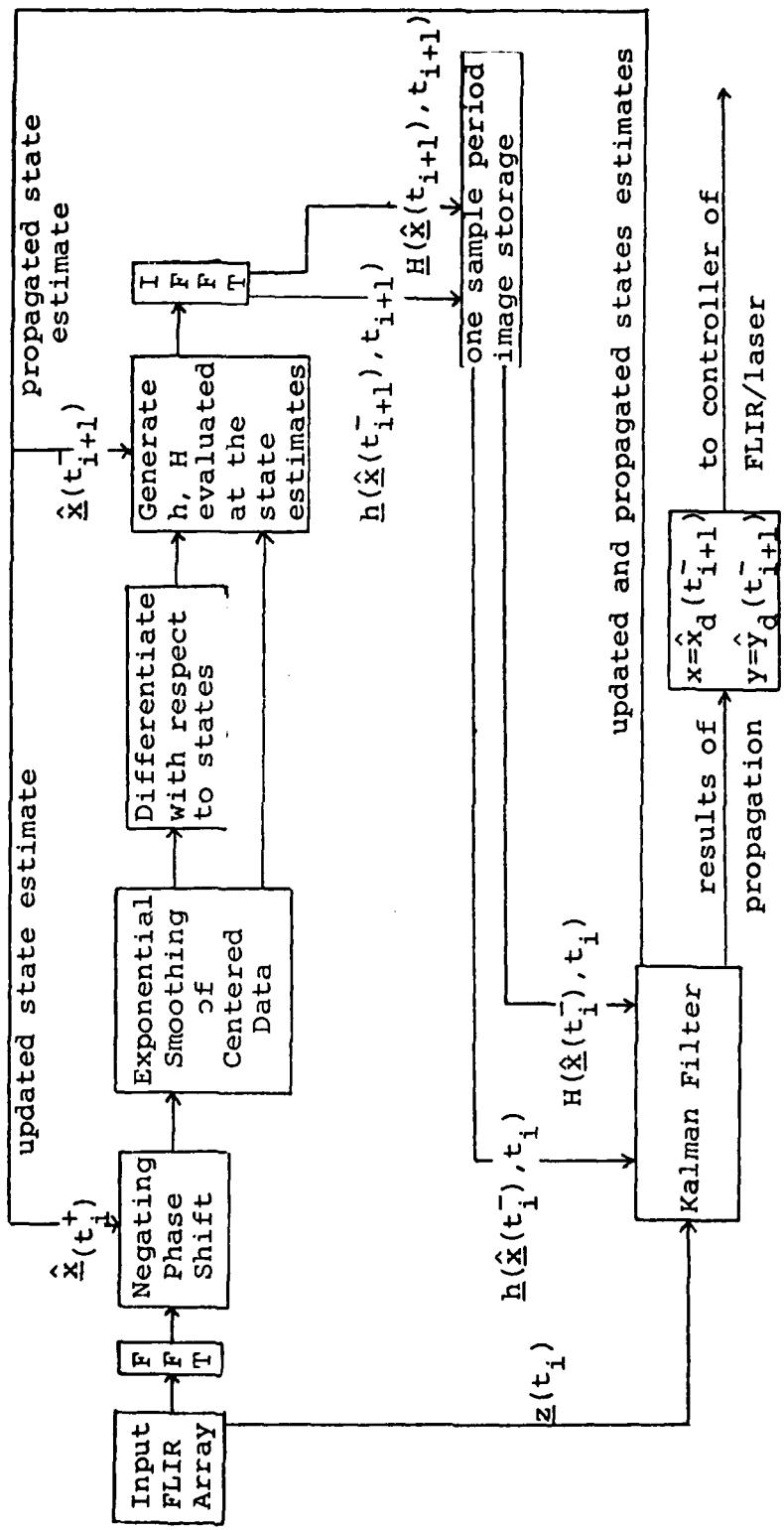


Figure 1. Data Processing Algorithm

The Kalman filter then uses its internal dynamics model to propagate its state estimates forward to produce its best estimate of the states at the next sample time, $\hat{x}(t_{i+1}^-)$. That information is passed to the controller, which will position the FLIR so that the center of its field-of-view is where the filter predicts the dynamics will place the target position at that next sample time. The upper path of Figure 1 is designed to provide the linearized and nonlinear intensity function for use by the Kalman filter.

The nonlinear intensity pattern, $h(\hat{x}(t_i^-), t_i)$, is the noise-free intensity measurement frame expected if the Kalman filter's propagated state estimates were perfect. Since the center of the field-of-view is equal to the filter-predicted target location due to the controller action, the nonlinear $h[\hat{x}(t_i^-), t_i]$ will be the target's intensity profile, h , offset from the center of the FLIR's field-of-view by the predicted atmospheric states. This is because the x-axis position of the centroid of the target's intensity profile is defined by

$$x_{\text{centroid}} = x_{\text{dynamics}} + x_{\text{atmospherics}} \quad (1-2)$$

and the control action is specifically intended to zero out the estimated x_{dynamics} , and similarly for the y-axis. The linearized intensity pattern, $H[\hat{x}(t_i^-), t_i]$, is also used by the Kalman filter, within the computation of the gain matrix $K(t_i)$, to incorporate a measurement to produce an updated

state estimate, $\hat{x}(t_i^+)$. This linearized intensity pattern is the derivative of the nonlinear intensity function, $h(\hat{x}(t_i^-), t_i)$, with respect to a change in the Kalman filter states evaluated at the predicted state at that sample time, $\hat{x}(t_i^-)$: $H[\hat{x}(t_i^-), t_i] = \partial h[\hat{x}(t_i^-), t_i] / \partial x$.

To generate these estimated target intensity patterns from the noise-corrupted FLIR requires interframe filtering to attenuate the noise. This interframe smoothing requires the target's intensity profile to be centered from frame to frame so that the noise can be averaged out. To center the target's intensity function within a data array which represents a FLIR's field-of-view, the shifting theorem of the Fourier Transform is used.

The offset of the target's intensity pattern from the center of the FLIR's field-of-view can be negated. This is accomplished by multiplying the Fourier Transform of the frame of data by the complex conjugate of the linear phase shift induced in the frequency domain by the translation of the intensity pattern from the center of the frame in the space domain. Atmospheric jitter and imperfect propagation of dynamic states are the sources of the image's translation from the center of the sensor's field-of-view in the space domain. The output of the Kalman filter's update Equation (1-1) provides an estimate of both of these offsets for the present frame of data, $\hat{x}(t_i^+)$. These estimates of the updated

target location and atmospheric jitter, $\hat{x}(t_i^+)$, are used to compute the estimated centroid location, as shown by Equation (1-2). The centroid location is used in the argument of the complex conjugate of the linear phase shift which negates the spatial translation effects and provides a centered intensity function. This centered pattern is then averaged with previously centered frames of data, using exponential smoothing, to reduce the effects of noise corruption. The derivative property of the Fourier Transform is then used to differentiate the centered smoothed intensity function with respect to a change in Kalman filter states.

This shape function and its derivatives are then evaluated at the state expected at the next sample time. For this application, where the FLIR is centered on the predicted target location, the intensity patterns are evaluated at the predicted atmospheric states. This is the intensity pattern which would be received if the image were noise-free and the Kalman filter had made perfect estimates of the dynamic and atmospheric states, of Equation (1-2), which determine the location of the centroid of the intensity pattern. The inverse Fourier transform is then computed and $h[\hat{x}(t_{i+1}^-), t_{i+1}]$ and $\underline{h}[\hat{x}(t_{i+1}^-), t_{i+1}]$ are ready for the Kalman filter to process the next frame of data.

Plan of Attack

This section provides an overview of the general flow

of this research, as pictured schematically in Figure 2.

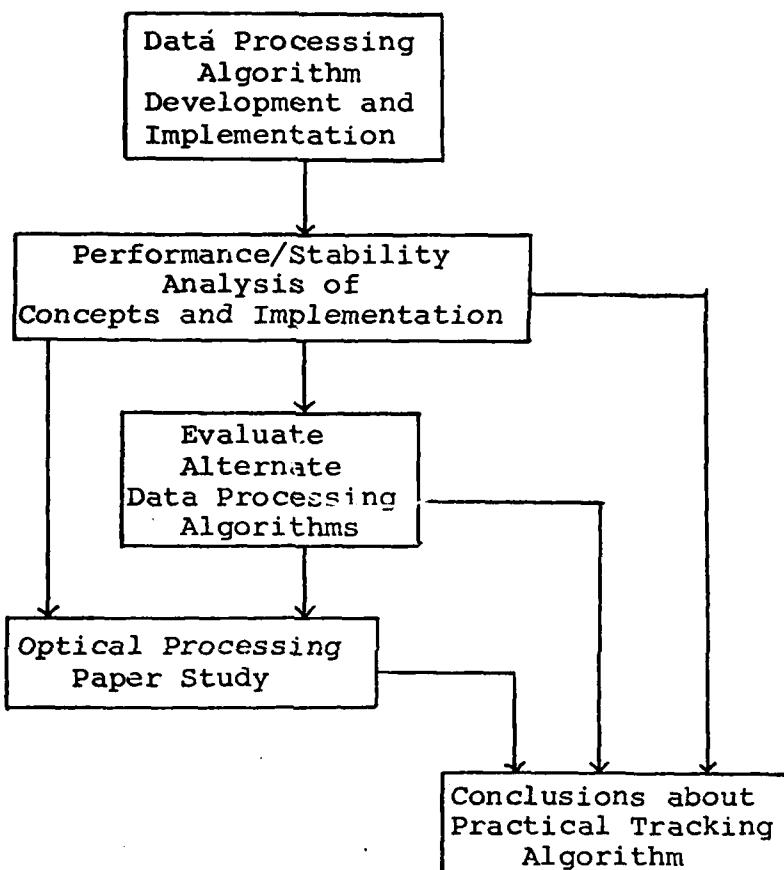


Figure 2. Research Plan of Attack Flow Diagram

The first block of Figure 2 represents the first step of this research. This step includes the development of the algorithms necessary to process the FLIR measurements. The complete implementation of Figure 1 was accomplished during this portion of the research. This included the development of the four-state extended Kalman filter needed to estimate

target position along with the estimated atmospheric disturbance in the x and y directions. To generate the intensity functions $\underline{h}[\hat{x}(t_i^-), t_i]$ and $\underline{H}[\hat{x}(t_i^-), t_i]$, algorithms were needed to take the forward/inverse two-dimensional Discrete Fourier Transform of the FLIR data. Once in the frequency domain, the capability to shift the data to center it within the frame was developed so interframe filtering could be accomplished. An exponential smoothing algorithm was developed for filtering out background noise. The derivative of this smoothed intensity function was derived via the derivative property of the Fourier transform. The shifting algorithm is then used to compute what the intensity function and its derivative would be as evaluated at the filter-predicted state at the next sample time.

The next objective of this research was to perform an analysis of the target tracking (and shape function generation secondarily) performance and stability of the data processing algorithm of Figure 1. Monte Carlo analysis techniques were used to gather statistics on the ability of this data processing algorithm to generate $\underline{h}[\hat{x}(t_i^-), t_i]$ and $\underline{H}[\hat{x}(t_i^-), t_i]$. It is the target tracking performance which is of primary concern.

The alternate data processing algorithm evaluation section evaluates the potential of an improved correlator tracker followed by a Kalman filter. The correlator tracker

works well in high signal-to-noise ratio scenarios and provides good estimates of centroid location, especially for extended targets (as opposed to point source targets). The Kalman filter could then filter these estimates, treated as measurements for the filter, to separate out the components of Equation 1-2.

The demand for real-time operation of the computationally intense algorithms discussed above requires the development of parallel processing techniques such as optical processing. The final section is a study into the potential of optically computing some of the quantities of Figure 1.

Overview

Chapters II, III, and IV will describe in detail the mathematical models used in the computer implementation. Chapter II presents the algorithms used in the derivation of the nonlinear and linearized intensity functions. Chapter III presents the mathematical model used for the truth model which represents the environment from which measurements are taken. Chapter IV describes the Kalman filter model used in the computer simulation. Chapter V discusses the possibility of a correlator followed by a Kalman filter to provide the position estimates. Chapter VI presents the results from the performance analysis which includes the statistics on how well the intensity functions are being constructed as well as statistics on target tracking ability for both

algorithms, and Chapter VII presents the optical processing options. Finally, Chapter VIII will present the conclusions and recommendations.

III. Derivation of the Nonlinear and Linearized Intensity Functions

Introduction

This chapter will present the algorithms necessary to produce an accurate reference intensity profile, $\underline{h}[\hat{x}(t_i^-), t_i]$, and the derivatives of that function, $\underline{H}[\hat{x}(t_i^-), t_i]$, with respect to the states. These intensity functions are needed by the extended Kalman filter to process FLIR measurements to update state estimates (see Chapter IV, Extended Kalman Filter). The true intensity profile is a vector of scalar components equal to the average intensity of the target's IR image over a particular pixel in the field-of-view. Noise-corrupted FLIR data must be processed to determine the target's profile. Figure 1 showed the data processing necessary to produce the intensity functions from the incoming data. The goal is to recognize the intensity function which is common to the noise-corrupted frames of data.

Pattern Recognition

All of the information of a two-dimensional intensity pattern can be preserved by a set of eigenvalues and eigenfunctions. To retain all of the information of a profile may require an infinite set of such values and functions. It is desirable to examine the FLIR image data in a coordinate system which has properties more conducive to recognizing patterns than the spatial x-y coordinate system. Ideally, a

linear transformation which rotates the FLIR image into a new coordinate space where the components are uncorrelated is desired. The Karhunen-Loeve Transformation is an example of such an operation. This transformation does possess the disadvantage of having to produce the correlation matrix which is $N^2 \times N^2$ if the input square matrix is $N \times N$.

The Karhunen-Loeve Transformation possesses the property of providing a coordinate space with perfectly uncorrelated image components, but it is a difficult transformation to perform in its exact form. The Karhunen-Loeve Transformation does provide motivation for the use of the more familiar Fourier Transform which uses complex exponentials as eigenfunctions (4:12). Where, in the Karhunen-Loeve Transformation the eigenvalues correspond to actual variance statistics projected on coordinate axes of basis vectors in a space where the image data is uncorrelated, the Fourier Transform is a projection of the image along the basis vectors formed by complex exponentials (9:195).

Although the Fourier Transform does not provide the perfect decorrelation, it does possess a computational advantage which outweighs the compression efficiency of the Karhunen-Loeve Transformation (9:194). The Fourier Transform also possesses the advantage that the two-dimensional transform is separable and can be obtained by one-dimensional operations.

Two-Dimensional Fourier Transforms

The purpose of decomposing the information contained within the complicated image intensity data into a set of eigenvalues and eigenfunctions is to compress that information into a set of inputs which will make manipulations performed on the data easier. Fourier analysis provides one way of performing such a decomposition.

The Fourier Transform of a complex-valued function $\tilde{g}(x, y)$ of two independent variables is a decomposition into a linear combination of elementary functions of the complex exponential form $\exp[j2\pi(f_x x + f_y y)]$ (7:8). This Fourier Transform is defined by

$$\tilde{G}(f_x, f_y) = F(\tilde{g}(x, y)) = \iint_{-\infty}^{\infty} \tilde{g}(x, y) \exp[-j2\pi(f_x x + f_y y)] dx dy \quad (2-1)$$

where

$\tilde{G}(f_x, f_y)$ = Frequency Spectrum

$\tilde{g}(x, y)$ = Spatial Domain Representation

f_x, f_y = Spatial Frequencies

x, y = Spatial Variables

The transform is also a complex-valued function of two independent variables f_x and f_y , which are the spatial frequencies. The inverse Fourier Transform is defined by

$$\tilde{g}(x, y) = F^{-1}(\tilde{G}(f_x, f_y)) = \iint_{-\infty}^{\infty} \tilde{G}(f_x, f_y) \exp[+j2\pi(f_x x + f_y y)] df_x df_y \quad (2-2)$$

The complex-valued number $\tilde{G}(f_x, f_y)$ is a weighting factor that

is applied to the eigenfunction, which represents the coordinate axes of the Fourier domain, in order to synthesize $\tilde{g}(x, y)$.

The Fourier Transform is separable, so that the two-dimensional transform domain can be reached with one-dimensional operations. Implementation is accomplished by transforming first on the rows of the image, followed by the transformation along the columns.

The two-dimensional finite Discrete Fourier Transform of a two-dimensional periodic sequence, of period N in both directions, is the finite sum of complex exponentials in the form

$$\tilde{G}(f_x, f_y) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \tilde{g}(x, y) \exp[-j\frac{2\pi}{N}(xf_x + yf_y)] \quad (2-3)$$

The complex sequence of intensity values, $\tilde{g}(x, y)$ is discretized into an N-pixel by N-pixel array. The inverse Discrete Fourier Transform is defined by

$$\tilde{g}(x, y) = \frac{1}{N^2} \sum_{f_x=0}^{N-1} \sum_{f_y=0}^{N-1} \tilde{G}(f_x, f_y) \exp[j\frac{2\pi}{N}(f_x x + f_y y)] \quad (2-4)$$

The two-dimensional complex-valued $N \times N$ array, $\tilde{G}(f_x, f_y)$, which represents the two-dimensional finite Discrete Fourier Transform of the original FLIR data, is discretized into spatial frequency bins. Each component of the array represents one of the $N/2$ spatial frequencies or their conjugates. The inverse transform differs from the forward transform only in

the sign of the exponent which appears in the complex exponential within the summation of Equation (2-4). The transform, $\tilde{G}(f_x, f_y)$, will be periodic of period N as was the original sequence, $\tilde{g}(x, y)$. Appendix A contains a further interpretation of the two-dimensional Discrete Fourier Transform. It also contains details on how the two-dimensional Fourier Transform implementation was tested.

In summary, the two-dimensional Discrete Fourier Transform assumes a finite length two-dimensional sequence is one period of a two-dimensional periodic sequence which then can be decomposed into a linear combination of complex exponentials. See Appendix A for explicit details.

Shifting Property of Fourier Transforms

To generate the target intensity patterns from the noise-corrupted FLIR data, interframe filtering is required to attenuate the noise. Interframe smoothing requires that the target's intensity profile be centered from frame to frame. Successive centered frames of data can then be averaged to attenuate the corrupting noise. The shifting property of the Fourier Transform is used in conjunction with the filter's estimated location of the intensity profile to center the data.

A shift in the scalar spatial domain of a finite duration one-dimensional sequence can be interpreted as a rotation in the basic interval (9:103). That is, the shifting of

samples out of one side of the periodic sequence results in identical samples entering the interval at the other end.

This is sometimes called a cylindrical shift. If, for example, the Fourier Transform of $\tilde{g}(x)$ is defined by

$$F[\tilde{g}(x)] = \tilde{G}(\omega) \quad (2-5)$$

then the Fourier Transform of the shifted sequence is defined by

$$F[\tilde{g}(x-x_0)] = e^{-j\omega x_0} \tilde{G}(\omega) \quad (2-6)$$

In the case of a finite-area two-dimensional sequence, a translation of the function in the space domain introduces a linear phase shift in the frequency domain (9:9):

$$F[\tilde{g}(x-x_0, y-y_0)] = \tilde{G}(f_x, f_y) \exp(-j2\pi(f_x x_0 + f_y y_0)) \quad (2-7)$$

where $F[\tilde{g}(x, y)] = \tilde{G}(f_x, f_y)$

An interpretation of the translation by x_0 and y_0 in the space domain, analogous to the one-dimensional interpretation, would be a rotation of each column by x_0 samples and each row by y_0 samples (9:119).

The offset of the target's intensity pattern from the center of the FLIR's field-of-view can be negated to provide a centered image for smoothing. This is accomplished by multiplying the Fourier Transform of the frame of data, which contains the offset intensity profile, by the complex conjugate of the linear phase shift induced in the frequency domain. The centered intensity profile can be obtained from

the offset profile by

$$\tilde{g}(x, y) = F^{-1}\{F[\tilde{g}(x-x_o, y-y_o)] \exp[j2\pi(f_x x_o + f_y y_o)]\} \quad (2-8)$$

In summary, by using the Kalman filter's updated estimates of the location of the centroid within a frame of data, the centered intensity profile can be obtained. Appendix B contains the details of how the shift is implemented.

Smoothing in the Fourier Domain

The target's intensity profile is not directly available for observation. This is because each FLIR frame is corrupted by inherent FLIR errors and background noise. Background noise can vary from zero to high temporal and spatial correlations. The FLIR errors, such as thermal noise and dark current effects, can be modelled as temporally and spatially uncorrelated noise. The effects of these noises on target intensity profile shape generation must be minimized by some appropriate data processing techniques. Any chosen data processing technique must not assume the precise form of the corrupting noise because of the large range of correlation characteristics. However, it is appropriate to exploit the fact that the corrupting noise changes faster from sample period to sample period than the target's intensity pattern (4;5).

An exponential smoothing algorithm was chosen to combat the effects of the corrupting noise. The equation which

defines exponential smoothing is:

$$\hat{y}(t) = \alpha y(t) + (1-\alpha)\hat{y}(t-1) \quad (10:115) \quad (2-9)$$

where

$\hat{y}(t)$ = current averaged value

$y(t)$ = current data frame

$\hat{y}(t-1)$ = previous averaged data frame

α = smoothing constant $0 \leq \alpha \leq 1$

The smoothing constant α can vary anywhere from 0 to 1. The value of α determines how the current averaged data frame, $\hat{y}(t)$, responds to the current data frame, $y(t)$. If the target's intensity pattern is only changing slowly, the value of α should tend more toward zero than one so that more emphasis is placed on the previous averaged data but the present frame is still used.

In summary, $\hat{y}(t)$ contains information from all the past data frames. The initial data frames have less of an effect on the most recent averaged data frame, $\hat{y}(t)$, than do the most recent data, as is appropriate for a slowly changing target image. Appendix C contains the details of how this smoothing algorithm was implemented and how the algorithm can be expressed in a closed form.

Derivative Property of Fourier Transforms

As stated in Chapter I and to be shown explicitly in Chapter IV, the extended Kalman filter algorithm requires

the derivatives of the intensity function with respect to the states, $\underline{H}[\hat{x}(t_i^-), t_i] = \partial h(\hat{x}(t_i^-), t_i) / \partial \underline{x}$. In past research efforts (4:), this linearized intensity function was derived using a numerical approximation known as the Forward-Backward Difference Method. This method only uses the pixels just before and just after the pixel being computed in the direction the spatial derivative is being taken. It was for this reason that the Derivative Property of the Fourier Transform, which uses all the data array, was chosen as a better means of generation for this research.

Differentiation in the space domain just becomes a multiplication by $j2\pi(f_x + f_y)$ in the frequency domain. The derivative of the intensity function can be expressed in terms of the transform of the function

$$F\left[\frac{\partial h(x, y)}{\partial x}\right] = j2\pi f_x \cdot F[h(x, y)] \quad (2-10a)$$

$$F\left[\frac{\partial h(x, y)}{\partial y}\right] = j2\pi f_y \cdot F[h(x, y)] \quad (2-10b)$$

Appendix D provides explicit details on the implementation of this algorithm.

In summary, the derivatives of the intensity function with respect to the states are required by the extended Kalman filter algorithm. These derivatives can be expressed in terms of the transform of that intensity function using Equations (2-10a) and (2-10b). Appendix D contains the details of Subroutine Deriv.

Derivation of Intensity Functions: Summary

This section will review the algorithms used to produce an accurate reference intensity profile, $h[\hat{x}(t_i^-), t_i]$, and the derivatives of that function $H[\hat{x}(t_i^-), t_i]$. As stated in Chapter I, the extended Kalman filter algorithm requires these intensity functions for the update state equations, (see Chapter IV, Extended Kalman Filter).

Noise-corrupted FLIR data must be processed to determine the target's intensity profile. The upper path of Figure 1 showed the data processing necessary to produce the intensity functions from the incoming noise-corrupted FLIR data. The goal of the data processing is to recognize the intensity function which is common to the noise-corrupted frames of data.

Interframe smoothing is required to generate the estimated target intensity pattern from the FLIR data. This smoothing is accomplished in the frequency domain since that is where the shift is accomplished, so the two-dimensional transform is first taken of the data. It will be shown in Appendix C on exponential smoothing that smoothing in the space domain is equivalent to smoothing in the frequency domain (see Table I, Chapter VI). To average from frame to frame, the target's intensity profile must be centered. This is accomplished via use of the Shifting Theorem of the Fourier Transform. Once the data is centered, exponential

smoothing is used to attenuate the noise while still in the frequency domain. The spatial derivatives are then obtained from this smoothed data by using the derivative property of the Fourier Transform. The shifting property can then be used again to derive the intensity functions evaluated at the predicted states. The high energy laser will be pointed at the best estimate of the target's position as calculated by the Kalman filter. The target's intensity function will be offset by atmospheric jitter from that target location. Since the field-of-view is centered at the estimated target location, the algorithm expects the intensity function to be offset from the center of the field-of-view by the estimated atmospheric jitter. By using the shifting property the estimated centered target intensity functions can be manipulated to produce the expected offset intensity functions.

In summary, this chapter presented the algorithms used to produce the intensity functions needed by the extended Kalman filter algorithm. The use of this procedure eliminates the need to make assumptions on the target's intensity profile.

III. Truth Model

Introduction

The truth model is the best mathematical representation of the real world environment from which the measurement vector, $\underline{z}(t_i)$, is generated. The environmental characteristics which are modelled include the underlying target dynamics, the atmospheric jitter effects, and the background and FLIR noises. This chapter presents the models used, along with an explanation of how the measurement vector, $\underline{z}(t_i)$, was created. The first two sections of this chapter present the two processes which contribute to movement of the intensity function, target dynamics and atmospheric jitter. These models are then transformed into state space notation and the propagation equations are given. Information on the computer creation of the measurement data for the computer simulation is next, with the final section containing the model for spatial correlations of background noise.

Target Dynamics Model

For this research, both deterministic and stochastic dynamic models were used. In the deterministic model, a target moving across the image plane at a constant velocity was simulated. In the stochastic model, a first order Gauss-Markov process was used to describe the movement of the target with respect to the field-of-view of the sensor. Although

there exist better models for describing the movement of specific targets, the very general stochastic model is applicable to many general targets. Together, these two models yield some of the basic characteristics of trajectories.

The deterministic model was developed to test the algorithm of Figure 1 with a target which is maintaining a constant velocity across the FLIR sensor's image plane. In this project, a constant velocity of three pixels over any twenty-frame time history was used.

$$\begin{aligned}x_d(t_{i+1}) &= x_d(t_i) + .15 \\y_d(t_{i+1}) &= y_d(t_i) + .15\end{aligned}\quad (3-1)$$

The stochastic model developed in Mercier's thesis, (5:9-10), for target dynamics was also used. In this model, the output of a first order linear shaping filter driven by zero mean white Gaussian noise was used to describe the movement of the target with respect to the FLIR's field-of-view. The outputs of these first order shaping filters, which describe the target dynamics in each direction, can be expressed as

$$\dot{x}_d(t) = \frac{-1}{T_t} x_d(t) + w_1(t) \quad (3-2)$$

$$\dot{y}_d(t) = \frac{-1}{T_t} y_d(t) + w_2(t) \quad (3-3)$$

where

$$E[w_1(t)] = E[w_2(t)] = 0 \quad (3-4)$$

$$E[w_1(t)w_1(s)] = E[w_2(t)w_2(s)] = \frac{2\sigma_d^2}{T_t} \delta(t-s) \quad (3-5)$$

$$E[w_1(t)w_2(s)] = 0 \quad (3-6)$$

and

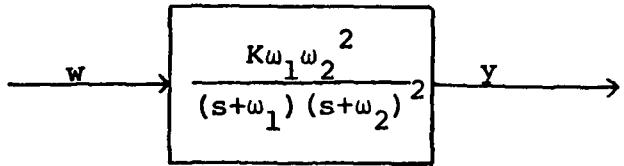
T_t = target correlation time

$w_1(t), w_2(t)$ = continuous time, independent, white Gaussian noise processes

σ_d^2 = desired variance on outputs x_d and y_d

Atmospheric Jitter Model

This section presents the mathematical model of the translational position changes of the target's intensity distribution due to disturbances in the atmosphere causing phase distortions in the radiated wavefronts. The model used is based on a study accomplished by The Analytic Sciences Corporation (11:29-30) and data analysis by Hogge and Butts (12:). As a part of these studies, a third order shaping filter driven by white Gaussian noise was developed to generate an output with a power spectral density representation that well approximated that of the effects of atmospheric turbulence. The jitter of the target intensity in each independent direction is modelled as the output of the third order shaping filter shown in Figure 3 (5:12). This filter has a single pole at 14.14 rad/sec and a double pole at 659.5 rad/sec, and it is driven by white Gaussian noise that is zero mean and has a strength of unity:



where

w = zero mean, unit strength, white Gaussian noise process

ω_1, ω_2 = break frequencies ($\omega_1 = 14.14$ rad/sec,
 $\omega_2 = 659.5$ rad/sec)

y = output of system

K = system gain

Figure 3. Third Order Shaping Filter

$$E[w(t)] = 0 \quad (3-7)$$

$$E[w(t) w(s)] = 1\delta(t-s) \quad (3-8)$$

As shown by Mercier, (5), by adjusting the value of the system gain, K, the desired root mean squared (RMS) atmospheric jitter characteristic is achieved. Appendix C of Mercier's thesis contains the details of this derivation.

State Space Model

The mathematical models developed in this chapter will now be transformed into state space notation. This transformation results in a time invariant vector differential equation of the form

$$\dot{\underline{x}}_t(t) = \underline{F}_t \underline{x}_t(t) + \underline{G}_t \underline{w}_t(t) \quad (3-9)$$

where

\underline{F}_t = truth model plant matrix

$\underline{x}_t(t)$ = truth model state vector

\underline{G}_t = truth model noise input matrix

\underline{w}_t = vector of white Gaussian noise inputs

$$E[\underline{w}_t(t)] = 0 \quad (3-10)$$

$$E[\underline{w}_t(t) \underline{w}_t^T(t + \tau)] = \underline{\Omega}_t \delta(\tau) \quad (3-11)$$

The truth model state vector is defined by

$$\underline{x}_t(t) = \begin{bmatrix} x_d(t) \\ x_{1a}(t) \\ x_{2a}(t) \\ x_{3a}(t) \\ y_d(t) \\ y_{1a}(t) \\ y_{2a}(t) \\ y_{3a}(t) \end{bmatrix} \quad (3-12)$$

If the Jordan canonical form, as developed by Mercier, is used, the plant matrix of Equation (3-9) becomes

$$F_t = \begin{bmatrix} -\frac{1}{T_t} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\omega_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\omega_2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\omega_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{-1}{T_t} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\omega_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\omega_2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\omega_2 \end{bmatrix}$$

$$\omega_1 = 14.14 \text{ rad/sec} \quad \omega_2 = 659.5 \text{ rad/sec} \quad (3-13)$$

The truth model noise matrix (5:11-14) is defined by

$$G_t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & G_1 & 0 & 0 \\ 0 & G_2 & 0 & 0 \\ 0 & G_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & G_1 \\ 0 & 0 & 0 & G_2 \\ 0 & 0 & 0 & G_3 \end{bmatrix} \quad (3-14)$$

$$G_1 = \frac{K\omega_1\omega_2^2}{(\omega_1-\omega_2)^2} \quad G_2 = -G_1 \quad G_3 = \frac{K\omega_1\omega_2^2}{(\omega_1-\omega_2)}$$

Using Equations (3-5) and (3-8), Ω_t of Equation (3-11) becomes

$$\Omega_t = \begin{bmatrix} \frac{2\sigma_d^2}{T_t} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{2\sigma_d^2}{T_t} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3-15)$$

The movement of the target's intensity profile, the centroid, is governed by components of these truth model matrices where

$$\begin{aligned} x_{peak}(t) &= x_d(t) + x_a(t) \\ y_{peak}(t) &= y_d(t) + y_a(t) \end{aligned} \quad (3-16a)$$

$$\begin{aligned}
 x_d(t) &= x_t(1) \\
 x_a(t) &= x_t(2) + x_t(3) \\
 y_d(t) &= x_t(5) \\
 y_a(t) &= x_t(6) + x_t(7)
 \end{aligned} \tag{3-16b}$$

Thus, the location of the centroid of the intensity profile can be expressed as

$$\begin{bmatrix} y_t(t) \\ y_{peak}(t) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_t(t) \end{bmatrix} \tag{3-17}$$

Propagation Equations

Propagation equations describe how the state vector makes the transition from one sample time to another. The solution to Equation (3-9) (5:14) is

$$\underline{x}_t(t_{i+1}) = \underline{\Phi}_t(t_{i+1}, t_i) \underline{x}_t(t_i) + \int_{t_i}^{t_{i+1}} \underline{\Phi}_t(t_{i+1}, \tau) \underline{G}_t(\tau) \underline{w}_t(\tau) d\tau \tag{3-18}$$

where $\underline{\Phi}_t(t, t_i)$ is the solution to the matrix differential equation

$$\dot{\underline{\Phi}}_t(t, t_i) = \underline{F}_t \underline{\Phi}_t(t, t_i) \tag{3-19a}$$

and initial condition

$$\underline{\Phi}_t(t_i, t_i) = \underline{I} \text{ (identity matrix)} \tag{3-19b}$$

The matrix $\underline{\Phi}_t(t_{i+1}, t_i)$ is the state transition matrix which describes the truth model state's transition from time t_i to

time t_{i+1} . The truth model plant matrix, \underline{F}_t of Equation (3-14), is a constant matrix, and therefore, the state transition matrix is only a function of the time difference $(t_{i+1} - t_i)$. Therefore, the state transition matrix becomes a constant for a fixed sample time $\Delta t = t_{i+1} - t_i$. The independent channels reflected in the truth model plant matrix result in the state transition matrix being block diagonal with both of the 4-by-4 diagonal blocks equal to

$$\underline{\Phi}_{t_x}(t_{i+1}, t_i) = \underline{\Phi}_{t_y}(t_{i+1}, t_i) = \begin{bmatrix} e^{-\Delta t/T_t} & 0 & 0 & 0 \\ 0 & e^{-\omega_1 \Delta t} & 0 & 0 \\ 0 & 0 & e^{-\omega_2 \Delta t} & \Delta t e^{-\omega_2 \Delta t} \\ 0 & 0 & 0 & e^{-\omega_2 \Delta t} \end{bmatrix} \quad (3-20)$$

where $\Delta t = \text{constant sampling time } (t_{i+1} - t_i)$. A probabilistic interpretation of the integral term of Equation (3-18) is required to describe the process $\underline{x}_t(t)$. This integral has a zero mean Gaussian distribution. For notational convenience the contribution of the integral is set equal to $\underline{w}_{td}(t_i)$.

$$\underline{w}_{td}(t_i) \stackrel{\Delta}{=} \int_{t_i}^{t_{i+1}} \underline{\Phi}_t(t_{i+1}, \tau) \underline{G}_t(\tau) \underline{w}_t(\tau) d\tau \quad (3-21)$$

Computing the statistics of $\underline{w}_{td}(t_i)$ (5:15) results in

$$E[\underline{w}_{td}(t_i)] = 0 \quad (3-22)$$

$$E[\underline{w}_{td}(t_i)\underline{w}_{td}^T(t_i)] = \int_{t_i}^{t_{i+1}} \underline{\Phi}_t(t_{i+1}, \tau) \underline{G}_t(\tau) \underline{\Omega}_t(\tau) \underline{G}_t^T(\tau) \underline{\Phi}_t^T(t_{i+1}, \tau) d\tau \quad (3-23)$$

$$E[\underline{w}_{td}(t_i)\underline{w}_{td}^T(t_j)] = 0 \quad (t_i \neq t_j) \quad (3-24)$$

The equivalent discrete-time model (5:15) for the above equations is given by Mercier to be

$$\underline{x}_t(t_{i+1}) = \underline{\Phi}_t(\Delta t) \underline{x}_t(t_i) + \sqrt{\underline{\Omega}_{td}} \underline{w}_n(t_i) \quad (3-25)$$

where $\sqrt{\underline{\Omega}_{td}}$ is the Cholesky square root (13:370) of

$$\underline{\Omega}_{td} \stackrel{\Delta}{=} \int_{t_i}^{t_{i+1}} \underline{\Phi}_t(t_{i+1}, \tau) \underline{G}_t(\tau) \underline{\Omega}_t(\tau) \underline{G}_t^T(\tau) \underline{\Phi}_t^T(t_{i+1}, \tau) d\tau \quad (3-26)$$

and

$$E[\underline{w}_n(t_i)] = 0 \quad (3-27)$$

$$E[\underline{w}_n(t_i)\underline{w}_n^T(t_j)] = I \delta_{ij} \quad (3-28)$$

The lower triangular Cholesky square root of $\underline{\Omega}_{td}$ satisfies the equation

$$\sqrt{\underline{\Omega}_{td}} \sqrt{\underline{\Omega}_{td}}^T = \underline{\Omega}_{td} \quad (3-29)$$

and can be uniquely defined for a given $\underline{\Omega}_{td}$.

Measurements

A measurement history of FLIR data frames is used by the Kalman filter to predict the pointing direction for the high energy laser. The output of an arbitrary pixel, say the $k-l$ -th pixel, within the FLIR data frame is the average

IR intensity over that pixel as sensed by a detector, and is described mathematically as

$$z_{kl}(t_i) = \frac{1}{A_p} \iint_{k-l-th \text{ pixel}} I_{target}(x, y, x_{peak}(t_i), y_{peak}(t_i)) dx dy + v_{kl}(t_i) \quad (3-30a)$$

where

I_{target} = the intensity of the target as a function of the location in the frame and the centroid peak as defined by Equation (1-2)

$z_{kl}(t_i)$ = the output of the $k-l$ -th pixel at a time t_i

A_p = the area of one pixel

(x, y) = coordinates of any point in the $k-l$ -th pixel

$(x_{peak}(t_i), y_{peak}(t_i))$ = the center of the Gaussian intensity pattern

$v_{kl}(t_i)$ = the additive noise for the $k-l$ -th pixel corresponding to background and FLIR noises.

in the general case. Using a bivariate Gaussian target intensity profile with circular equal intensity contours as a special case (4:25), the mathematical description becomes

$$\begin{aligned} z_{kl}(t_i) &= \frac{1}{A_p} \iint_{k-l-th \text{ pixel}} I_{max} \exp\left[\frac{-1}{2\sigma_g^2} [(x-x_{peak}(t_i))^2 \right. \\ &\quad \left. + (y-y_{peak}(t_i))^2]\right] dx dy + v_{kl}(t_i) \end{aligned} \quad (3-30b)$$

where

I_{max} = the maximum intensity received from target

σ_g^2 = the dispersion of the Gaussian intensity function

This would be the measurement vector for a single Gaussian

hotspot target. For simulation of the multiple hotspot targets of interest for this research, three exponentials of Gaussian form are summed in place of the single Gaussian of Equation (3-30a).

Spatially Correlated Background Noise

The existence of spatial correlations in the background of each FLIR data frame has been documented by Harnly and Jensen (6:14). To discuss the generation of the spatially correlated noise, a numbering system for each pixel of an 8 x 8 input array is adopted (Figure 4). The 8 x 8 FLIR data array is numbered, as done by Harnly and Jensen (6:19) and Singletary (4:21) by pixels from 1 to 64 starting in the upper left-hand corner and proceeding across the rows.

In the 64 x 64 correlation coefficient matrix corresponding to a noise vector with each component associated with a single such pixel, each element relates the noise in that pixel with the noise in another pixel. For this research the spatial correlations are modelled as extending to the first and second nearest pixels. The element in row 1 and column 1 of the correlation coefficient matrix relates the noise in pixel one to itself and is therefore one. Similarly, the entire diagonal of the correlation coefficient matrix is equal to one. The correlation coefficient matrix is symmetric since the correlation r_{ij} of the noise in pixel i with noise of pixel j is the same as that of the noise in

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

Figure 4. FLIR Data Frame Pixel Numbering Scheme (6:19)

pixel j with the noise in pixel i ($r_{ij} = r_{ji}$).

The exponentially decaying and radially symmetric form of the correlation function was used in generating the coefficients of the first and second nearest neighbor correlation matrix. Equation (3-31), was used to generate the coefficient matrix:

$$r_{ij} = \exp(-d_{ij}) \quad (3-31)$$

where d_{ij} = distance in # pixels from pixel i to pixel j. Equation (3-31) implements a correlation distance of one pixel. The fact that nonzero values are computed only for the first and second neighbors results in only the coefficients relating a pixel's noise to its 24 nearest neighbors being nonzero (Figure 5).

$r_{k,k-18} = .0591$	$r_{k,k-17} = .1069$	$r_{k,k-16} = .1353$	$r_{k,k-15} = .1069$	$r_{k,k-14} = .0591$
$r_{k,k-10} = .1069$	$r_{k,k-9} = .2431$	$r_{k,k-8} = .3679$	$r_{k,k-7} = .2431$	$r_{k,k-6} = .1069$
$r_{k,k-2} = .1353$	$r_{k,k-1} = .3679$	$r_{k,k} = 1$	$r_{k,k+1} = .3679$	$r_{k,k+2} = .1353$
$r_{k,k+6} = .1069$	$r_{k,k+7} = .2431$	$r_{k,k+8} = .3679$	$r_{k,k+9} = .2431$	$r_{k,k+10} = .1069$
$r_{k,k+14} = .0591$	$r_{k,k+15} = .1069$	$r_{k,k+16} = .1353$	$r_{k,k+17} = .1069$	$r_{k,k+18} = .0591$

Figure 5. First and Second Nearest Neighbor

In Figure 5, $r_{k,\ell}$ is the coefficient relating noise components in pixel k to those of pixel ℓ . Of the 25 nonzero values for any row or column, only six of them are distinct; one value of 1 for the correlation of a pixel with itself, four values of .3679 for pixels one pixel away, four values of .2431 for pixels $\sqrt{2}$ pixels away, four values of .1353 for pixels two pixels away, eight values of .1069 for pixels $\sqrt{5}$ pixels away, and four values of .0591 for pixels $\sqrt{8}$ pixels away. The matrix of correlation coefficients becomes

$$\underline{\underline{r}} = \begin{bmatrix} 1 & r_{1,2} & r_{1,3} & \dots & r_{1,64} \\ r_{1,2} & 1 & r_{2,3} & \dots & r_{2,64} \\ r_{1,3} & r_{2,3} & 1 & \dots & r_{3,64} \\ \dots & \dots & \dots & \dots & \dots \\ r_{1,64} & r_{2,64} & r_{3,64} & \dots & 1 \end{bmatrix}$$

Using the simplified noise covariance matrix of Harnly and Jensen, (6:18), the process for completing the generation of the noise covariance matrix is to premultiply the correlation coefficient matrix by the variance of the background noise.

$$\underline{R} = \sigma_b^2 \quad \underline{\underline{r}} = \sigma_b^2 \begin{bmatrix} 1 & r_{1,2} & r_{1,3} & \cdots & r_{1,64} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ r_{1,64} & r_{2,64} & r_{3,64} & \cdots & 1 \end{bmatrix}$$

Once the correlation matrix, \underline{R} is known, specific spatially correlated realizations of the 64×1 noise vector, $\underline{v}(t_i)$ can be computed by using a Cholesky square root decomposition of \underline{R} and a zero mean unit-variance independent Gaussian noise generator (see Appendix E). The noise vector realization will be added to the 8×8 input array. This noise vector is created by

$$\underline{v}(t_i) = \underline{\underline{C}} \underline{v}'(t_i) \quad (3-32)$$

where

$\underline{v}'(t_i)$ = 64 dimensional white Gaussian vector; each component zero mean, unit variance, and independent of other components (see Appendix E)

$\underline{\underline{C}}$ = Cholesky square root

The Cholesky square root decomposition (Chapter 7 reference 13) produces a lower triangular matrix such that

$$\underline{\underline{C}} \quad \underline{\underline{C}}^T = \underline{R} \quad (3-33)$$

By using this formulation of the noise vector, Equation (3-32), the covariance of $\underline{v}(t_i)$ is shown to be equal to the correlation matrix.

$$\begin{aligned} E[\underline{v}(t_i)\underline{v}^T(t_j)] &= E[\sqrt{R} \underline{v}'(t_i) \underline{v}'^T(t_j) \sqrt{R}^T] \\ &= \sqrt{R} E[\underline{v}'(t_i) \underline{v}'^T(t_j)] \sqrt{R}^T \\ &= \sqrt{R} I \sqrt{R}^T \delta_{ij} \\ &= R \delta_{ij} \end{aligned} \tag{3-34}$$

Summary of Truth Model

This chapter presented the mathematical models used to account for target dynamics, atmospheric jitter effects, and the background and FLIR noises. These models described the translational position changes of the target's intensity distribution resulting from any of these reasons. The mathematical models were then summarized by transforming them into state space notation. Equation (3-25) showed how the truth model state space states are propagated forward in time while still accounting for the stochastic nature of the problem. Information on the computer creation of the measurement data for the computer simulation was also provided along with the model for spatial correlations of background noise.

IV. Extended Kalman Filter

Introduction

The previous chapter presented the truth model which is the best representation of the environment from which the measurement vector $\underline{z}(t_i)$ is generated. An extended Kalman filter is used to process these measurement vectors to provide an estimate of the true target centroid location for the next sample time (for use by the controller) as well as of current state vectors needed for centering in the intensity image processing. A four-state extended Kalman filter model which accounts for time-corrupted dynamics and the bandwidth characteristics of the atmospheric jitter, as developed by Mercier, was used for this research. A wide variety of targets may be represented with this model by choosing appropriate noise strengths and correlation times.

Target Dynamics and Atmospheric Jitter Filter Models

The target dynamics and the atmospheric jitter are modelled as stationary, first order Gauss-Markov processes. These processes are generated as the outputs of first-order lags driven by white Gaussian noise (5:19). The differential equation which describes any of these modelled states is

$$\dot{x}_f(t) = \frac{-1}{T_f} x_f(t) + w_f(t) \quad (4-1)$$

where

$$E[w_f(t)] = 0 \quad (4-2)$$

$$E[w_f(t)w_f(t+\tau)] = \frac{2\sigma_f^2}{T_f} \delta(\tau) \quad (4-3)$$

and

T_f = correlation time

$w_f(t)$ = white Gaussian noise process

σ_f = root mean squared value of x

The extended Kalman filter uses this stationary, first order, Gauss-Markov process to model the target dynamics and the atmospheric jitter in the x and y directions of the FLIR image plane. The stochastic differential equation model for each state upon which the filter is based is Equation (4-1). This is the same structure as Equations (3-2) and (3-3) except that a reduced-order model is used for atmospherics.

State Space Model

The four filter states will now be put into state space notation. The state vector differential equation which describes the four filter states $(x_d \ x_a \ y_d \ y_a)^T$, is

$$\dot{\underline{x}}_f(t) = \underline{F}_f \ \underline{x}_f(t) + \underline{w}_f(t) \quad (4-4)$$

where

\underline{F}_f = filter plant matrix

$\underline{x}_f(t)$ = filter state vector

$\underline{w}_f(t)$ = input white Gaussian noise vector

Using the information from the previous section Equation (4-4) becomes

$$\dot{\underline{x}}_f(t) = \begin{bmatrix} -\frac{1}{T_{df}} & 0 & 0 & 0 \\ 0 & \frac{-1}{T_{af}} & 0 & 0 \\ 0 & 0 & \frac{-1}{T_{df}} & 0 \\ 0 & 0 & 0 & \frac{-1}{T_{af}} \end{bmatrix} \underline{x}_f(t) + \underline{w}_f(t) \quad (4-5)$$

where

T_{df} = correlation time assumed for target dynamics

T_{af} = correlation time assumed for atmospheric jitter

$$E(\underline{w}_f(t)) = \underline{0} \quad (4-6)$$

$$E(\underline{w}_f(t)\underline{w}_f^T(t+\tau)) = \underline{\Omega}_f \delta(\tau) \quad (4-7)$$

and

$$\underline{\Omega}_f = \begin{bmatrix} \frac{2\sigma_{df}^2}{T_{df}} & 0 & 0 & 0 \\ 0 & \frac{2\sigma_{af}^2}{T_{af}} & 0 & 0 \\ 0 & 0 & \frac{2\sigma_{df}^2}{T_{df}} & 0 \\ 0 & 0 & 0 & \frac{2\sigma_{af}^2}{T_{af}} \end{bmatrix} \quad (4-8)$$

σ_{df}^2 = assumed target dynamics noise variance

σ_{af}^2 = assumed atmospheric jitter noise variance

State Propagation

The extended Kalman filter must propagate its state estimate vector and conditional covariance matrix from one sample time to the next. The extended Kalman filter model state equations are linear in this application, and so the standard Kalman filter propagation equations can be used to propagate the filter states between sample times. These standard propagation equations are

$$\hat{x}(t_{i+1}^-) = \underline{\Phi}_f(t_{i+1}, t_i) \hat{x}(t_i^+) \quad (4-9)$$

$$\begin{aligned} \underline{P}(t_{i+1}^-) &= \underline{\Phi}_f(t_{i+1}, t_i) \underline{P}(t_i^+) \underline{\Phi}_f^T(t_{i+1}, t_i) + \\ &\quad \int_{t_i}^{t_{i+1}} \underline{\Phi}_f(t_{i+1}, \tau) \underline{\Omega}_f \underline{\Phi}_f^T(t_{i+1}, \tau) d\tau \end{aligned} \quad (4-10)$$

where

$\underline{\Phi}_f(t_{i+1}, t_i)$ = filter state transition matrix

$\underline{P}(t_i^+)$ = conditional state covariance matrix from measurement update equation at time t_i

$\underline{P}(t_{i+1}^-)$ = conditional state covariance matrix propagated from time t_i to t_{i+1}

$\underline{\Omega}_f$ = noise covariance kernel descriptor given in Equation (4-8)

The filter's state transition matrix $\underline{\Phi}_f(t_{i+1}, t_i)$ must satisfy the differential equation

$$\dot{\underline{\Phi}}_f(t, t_i) = \underline{E}_f \underline{\Phi}_f(t, t_i) \quad (4-11)$$

over the interval (t_i, t_{i+1}) , subject to the initial condition

$$\underline{\Phi}_f(t_i, t_i) = \underline{I} \text{ (identity matrix)} \quad (4-12)$$

The time-invariant plant matrix, \underline{F}_f , and fixed sample period result in a constant state transition matrix, $\underline{\Phi}_f(t_{i+1}, t_i)$, for any given sample period:

$$\underline{\Phi}_f(t_{i+1}, t_i) = e^{\underline{F}(t_{i+1}-t_i)} = e^{\underline{F}\Delta t}$$

$$= \begin{bmatrix} e^{-\Delta t/T_{df}} & 0 & 0 & 0 \\ 0 & e^{-\Delta t/T_{af}} & 0 & 0 \\ 0 & 0 & e^{-\Delta t/T_{df}} & 0 \\ 0 & 0 & 0 & e^{-\Delta t/T_{af}} \end{bmatrix} \quad (4-13)$$

where Δt is the sample period, $\Delta t = t_{i+1} - t_i$.

The solution to the integral term of Equation (4-10) becomes

$$\underline{Q}_D = \begin{bmatrix} \sigma_{df}^2 (1-e^{-2\Delta t/T_{df}}) & 0 & 0 & 0 \\ 0 & \sigma_{af}^2 (1-e^{-2\Delta t/T_{af}}) & 0 & 0 \\ 0 & 0 & \sigma_{df}^2 (1-e^{-2\Delta t/T_{df}}) & 0 \\ 0 & 0 & 0 & \sigma_{af}^2 (1-e^{-2\Delta t/T_{af}}) \end{bmatrix} \quad (4-14)$$

The matrix of Equation (4-14) is constant for a given sampling time Δt , due to system time invariance and noise stationarity.

The values of σ_{df}^2 and σ_{af}^2 of the \underline{Q}_f matrix of Equation (4-8)

are determined during an off-line tuning process. This off-line tuning produces the values which optimize tracking performance.

In summary, the estimated state vector and the conditional covariance matrix propagation equations are

$$\hat{x}(t_{i+1}^-) = \begin{bmatrix} e^{-\Delta t/T_{df}} & 0 & 0 & 0 \\ 0 & e^{-\Delta t/T_{af}} & 0 & 0 \\ 0 & 0 & e^{-\Delta t/T_{df}} & 0 \\ 0 & 0 & 0 & e^{-\Delta t/T_{af}} \end{bmatrix} \hat{x}(t_i^+) \quad (4-15)$$

$$P(t_{i+1}^-) = \begin{bmatrix} e^{-\Delta t/T_{df}} & 0 & 0 & 0 \\ 0 & e^{-\Delta t/T_{af}} & 0 & 0 \\ 0 & 0 & e^{-\Delta t/T_{df}} & 0 \\ 0 & 0 & 0 & e^{-\Delta t/T_{af}} \end{bmatrix} P(t_i^+)$$

$$\cdot \begin{bmatrix} e^{-\Delta t/T_{df}} & 0 & 0 & 0 \\ 0 & e^{-\Delta t/T_{af}} & 0 & 0 \\ 0 & 0 & e^{-\Delta t/T_{df}} & 0 \\ 0 & 0 & 0 & e^{-\Delta t/T_{af}} \end{bmatrix} +$$

$$\begin{bmatrix} \sigma_{df}^2 (1-e^{-2\Delta t/T_{df}}) & 0 & 0 & 0 \\ 0 & \sigma_{af}^2 (1-e^{-2\Delta t/T_{af}}) & 0 & 0 \\ 0 & 0 & \sigma_{df}^2 (1-e^{-2\Delta t/T_{df}}) & 0 \\ 0 & 0 & 0 & \sigma_{af}^2 (1-e^{-2\Delta t/T_{af}}) \end{bmatrix} \quad (4-16)$$

Equations (4-15) and (4-16) propagate the four filter states between sample times. Once the predicted states for the next sample time are calculated, the estimates of the target's position $x_d(t_{i+1}^-)$ and $y_d(t_{i+1}^-)$, are fed to a feedback controller. The controller positions the FLIR to be centered on this estimated target location by that next sample time t_{i+1} . In view of Equation (3-16), the filter expects the apparent target's intensity function to be offset from the center of that field-of-view by the estimated jitter effects also calculated by Equation (4-15).

Measurement Update Equations

Equation (3-30) can be rewritten in the general form

$$z_{kl}(t_i) = h_{kl}(\underline{x}(t_i), t_i) + v_{kl}(t_i) \quad (4-17)$$

where $h_{kl}(\underline{x}(t_i), t_i)$ is a nonlinear function of the states at sample time t_i . Equation (4-17) represents each pixel's information as a nonlinear function of the states plus additive corrupting noise, $v_{kl}(t_i)$. Once the Kalman filter's

state vector and covariance matrix are propagated to the next sample time, the filter uses this propagated state estimate along with the information contained within each of the 64 pixels represented by Equation (4-17), to compute an updated estimate of the underlying target centroid location. To process the information contained within the measurement vector, the extended Kalman filter was used, since it can handle the nonlinearities of the measurement equation, Equation (4-17). Moreover, it is less computationally burdensome than other nonlinear filters, and computation time is of extreme importance in this application.

In the process of generating the appropriate filter gain, an extended Kalman filter linearizes the measurement equation about the most recent estimate of the states, $\hat{x}(t_i^-)$. The inverse covariance form (13:257) of the extended Kalman filter measurement update equations is used to eliminate the need for the inversion of a 64×64 matrix for every update (5:26). This 64×64 inversion would be required during the calculation of the Kalman filter gain in the usual form of the update Equations (13:233), i.e., $K = P^{-1}H^T(HP^{-1}H^T + R)^{-1}$, because of the 64 scalar measurements. The extended Kalman filter update equations in inverse covariance form are

$$P^{-1}(t_i^+) = P^{-1}(t_i^-) + H^T(t_i)R^{-1}(t_i)H(t_i) \quad (4-18)$$

$$P(t_i^+) = [P^{-1}(t_i^+)]^{-1} \quad (4-19)$$

$$K(t_i) = P(t_i^-)^T R^{-1}(t_i) \quad (4-20)$$

$$\hat{x}(t_i^+) = \hat{x}(t_i^-) + K(t_i) [z(t_i) - h(\hat{x}(t_i^-), t_i)] \quad (4-21)$$

where

$$H(t_i) = \frac{\partial h(\hat{x}(t_i^-), t_i)}{\partial \underline{x}} = \text{linearized function of intensity measurements (see Appendix F)} \quad (4-22)$$

$P(t_i^-)$ = propagated conditional covariance matrix before measurement incorporation at time t_i^-

$\hat{x}(t_i^-)$ = propagated state estimate vector of filter before measurement incorporation at time t_i^-

$K(t_i)$ = Kalman filter gain

$h(\hat{x}(t_i^-), t_i)$ = nonlinear function of intensity measurements at time t_i^- , as function of filter state estimates $\hat{x}(t_i^-)$

$z(t_i)$ = actual realization of measurement vector

This formulation of the extended Kalman filter update equations requires only two 4×4 inverses, of the $P(t_i^-)$ matrix and the $P^{-1}(t_i^+)$, and therefore eases the computational burden of the algorithm substantially. The inversion of the $R(t_i)$ matrix is accomplished only once, potentially offline, and the result of that inversion, $R^{-1}(t_i)$, is stored for use in Equation (4-18). Moreover, in the spatially uncorrelated pixel noise case, $R^{-1}(t_i)$ is of the form $(1/R)\mathbf{I}$.

In this research, the nonlinear h function and its derivatives with respect to the filter states are derived using the FFT, phase shifting and smoothing techniques dis-

cussed in Chapter II. The structure of this algorithm is shown in Figure 1.

In summary, the inverse covariance form of the extended Kalman filter measurement update equations were used in this research to ease the computational burden. Equations (4-18) through (4-21) present the equations implemented in the computer simulation.

Summary of the Extended Kalman Filter Equations

This section summarizes the propagation and update equations used for this research

Propagation Equations

$$\hat{x}(t_i^-) = \underline{\Phi}_f \hat{x}(t_{i-1}^+) \quad (4-23)$$

$$\underline{P}(t_i^-) = \underline{\Phi}_f \underline{P}(t_{i-1}^+) \underline{\Phi}_f^T + \underline{\Omega}_D \quad (4-24)$$

where

$$\underline{\Phi}_f = \begin{bmatrix} e^{-\Delta t/T_{df}} & 0 & 0 & 0 \\ 0 & e^{-\Delta t/T_{af}} & 0 & 0 \\ 0 & 0 & e^{-\Delta t/T_{df}} & 0 \\ 0 & 0 & 0 & e^{-\Delta t/T_{af}} \end{bmatrix} \quad (4-25)$$

and where

$$\underline{\Omega}_D = \begin{bmatrix} \sigma_{df}^2 (1 - e^{-2\Delta t / T_{df}}) & 0 & 0 & 0 \\ 0 & \sigma_{df}^2 (1 - e^{-2\Delta t / T_{af}}) & 0 & 0 \\ 0 & 0 & \sigma_{df}^2 (1 - e^{-2\Delta t / T_{df}}) & 0 \\ 0 & 0 & 0 & \sigma_{df}^2 (1 - e^{-2\Delta t / T_{af}}) \end{bmatrix} \quad (4-26)$$

Estimate of states as modified by controller action:

$$\hat{\underline{x}}'(t_i^-) = \begin{bmatrix} 0 \\ \hat{x}_a(t_i^-) \\ 0 \\ \hat{y}_a(t_i^-) \end{bmatrix} \quad (4-27)$$

Update Equations

$$\underline{P}^{-1}(t_i^+) = \underline{P}^{-1}(t_i^-) + \underline{H}^T(t_i) \underline{R}^{-1}(t_i) \underline{H}(t_i) \quad (4-28)$$

$$\underline{P}(t_i^+) = [\underline{P}^{-1}(t_i^+)]^{-1} \quad (4-29)$$

$$\underline{K}(t_i) = \underline{P}(t_i^+) \underline{H}^T(t_i) \underline{R}^{-1}(t_i) \quad (4-30)$$

$$\hat{\underline{x}}(t_i^+) = \hat{\underline{x}}'(t_i^-) + \underline{K}(t_i) [\underline{z}(t_i) - \underline{h}(\hat{\underline{x}}'(t_i^-), t_i)] \quad (4-31)$$

V. Correlator-Kalman Filter Tracker

Introduction

The Kalman filter operating upon the raw digitized image has been shown to perform well in comparison to standard correlation trackers (5;6). In these studies, the filter had valid apriori information about the analytic form of the intensity function, $h(\hat{x}(t_i), t_i)$. The performance of the Kalman filter operating upon the raw digitized data versus a correlator should be less beneficial to the filter without such apriori intensity function information. It is also computationally easier to implement a correlation algorithm than to implement a high measurement dimension extended Kalman filter.

This chapter presents an alternate tracking concept which is a hybrid in that it uses both correlation and Kalman filtering. A correlator is first used to generate position estimates for the target within a noise-corrupted frame of data. The correlator uses the estimated intensity function, $h(\hat{x}(t_i), t_i)$, generated via the data processing algorithm of Chapter II, as its template. This is different from current algorithms which just use previous raw data as a template. This modification to the standard correlation algorithm is expected to enhance performance. The standard correlation algorithm does not exploit any knowledge of the target's dynamics or any knowledge of those disturbances which could cause apparent translational intensity function offsets.

The improved correlation algorithm developed for this research positions the template with the benefit of $\hat{x}(t_{i+1}^-)$ which does exploit such knowledge. To account for correlation errors, the position estimates of the correlator are processed by a linear Kalman filter to produce better position estimates.

Section one of this chapter provides information on how the correlation algorithm was implemented. The next section statistically characterizes the errors in the position estimates of this implementation of the correlation algorithm. The last section of this chapter develops the linear Kalman filter which processes the correlator's position estimates.

Correlator Implementation

This section presents the implementation of the correlation algorithm used to provide position "measurements" to the Kalman filter. The correlation routine, written for this research, computes the cross correlation of a template, the result of the data processing of Chapter II, and the noise-corrupted FLIR data frame. The FFT is used to compute the cross correlation (8:557) as shown below.

$$F[g(x,y)] = G(f_x, f_y) \quad (5-1)$$

$$F[l(x,y)] = L(f_x, f_y) \quad (5-2)$$

$$F[g(x,y) * l(x,y)] = G(f_x, f_y) \cdot L^*(f_x, f_y) \quad (5-3)$$

where

$g(x,y) * \underline{L}(x,y)$ = cross correlation of the two-dimensional spatial sequences $g(x,y)$ and $\underline{L}(x,y)$

$\underline{L}^*(f_x, f_y)$ = complex conjugate of the Fourier transform of the sequence $\underline{L}(x,y)$

Taking the inverse transform of Equation (5-3) yields the cross correlation of the two-dimensional sequences $g(x,y)$ and $\underline{L}(x,y)$:

$$\underline{R}(x,y) = g(x,y) * \underline{L}(x,y) = F^{-1}[\underline{G}(f_x, f_y) \cdot \underline{L}^*(f_x, f_y)] \quad (5-4)$$

where $\underline{R}(x,y)$ is the result of the correlation.

To show how the position estimates are obtained from this algorithm, a simple two-dimensional Gaussian array is examined. Let $\underline{L}(x,y)$ represent the two-dimensional array which contains the template, which for this example is a perfect target replica and is a centered Gaussian function with $\sigma^2 = 2$ pixels within the two-dimensional array. Figure 6 is an example of a 24 x 24 pixel array with the template's intensity being represented by a gray-scale (see Subroutine Display Appendix H). Each dash around the perimeter of the field-of-view represents one pixel in Figure 6. The use of the 24 x 24 pixel array allows the 8 x 8 pixel tracking window to be padded by 8 rows and 8 columns of zeros or data for the computation of FFTs.

Similarly, let $g(x,y)$ represent the two-dimensional array of noise-corrupted data, which contains the target's

offset intensity function. Figure 7 is an example of a 24 x 24 pixel array which is slightly noise corrupted. The target's single Gaussian intensity pattern is offset by one pixel in the horizontal direction from the center of the 24 x 24 pixel array. A gray scale representation of the result of the cross correlation, $\underline{R}(x,y)$ is shown in Figure 8.

The expected result of correlating a Gaussian with a Gaussian is a Gaussian whose location within the resulting matrix, $\underline{R}(x,y)$, indicates the relative pixel position offset between the template and the target. The two-dimensional array $\underline{R}(x,y)$ is symmetric in the vertical direction which indicates no offset there. In the horizontal direction a one pixel circular shift to the left would restore symmetry. That is, if all of the columns of Figure 8 were shifted one column to the left and the leftmost column shifted into the rightmost column, horizontal symmetry would be restored. Obviously the information that the target is one pixel horizontally offset from the center of the two-dimensional noise-corrupted data array is contained in the result of the cross correlation, $\underline{R}(x,y)$. The fact that the two-dimensional Fourier transform assumes that this two-dimensional sequence is one period in each direction of an infinitely periodic sequence explains the use of the circular shift. Inherent in the use of the FFT to derive $\underline{R}(x,y)$ is the assumption that the next pixel to the left of any pixel in column one

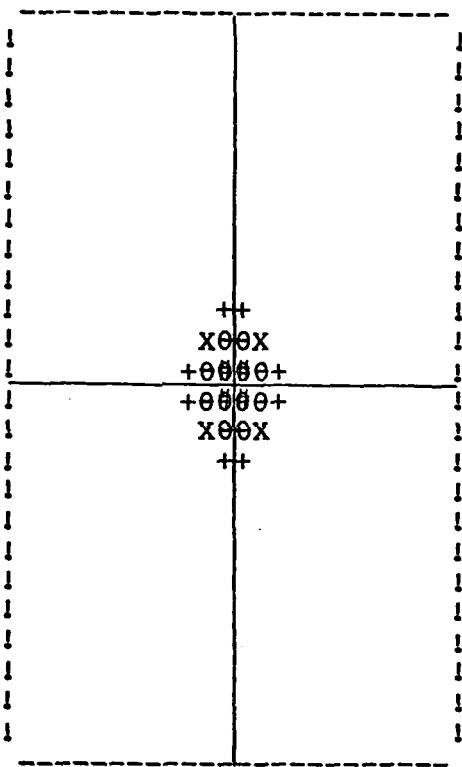


Figure 6. Centered Single Gaussian Template

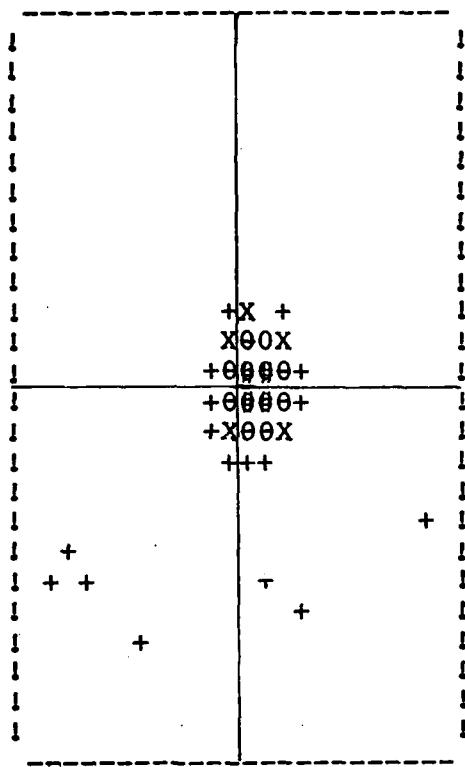


Figure 7. Noise Corrupted Data Array

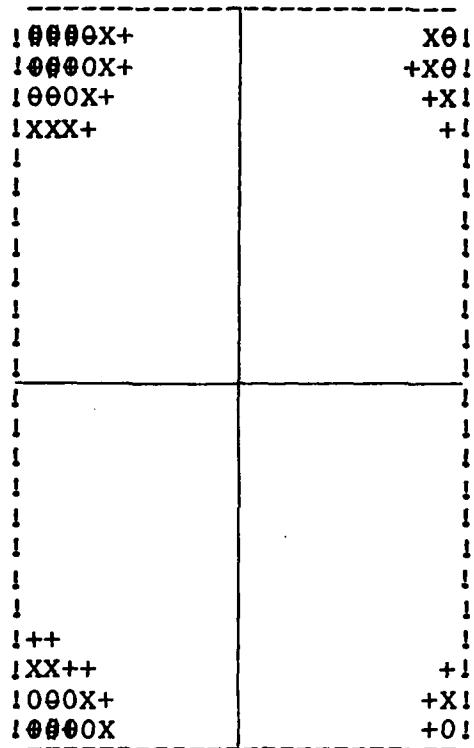


Figure 8. Result of Cross Correlation of Data and Template

is equal to the corresponding pixel in column 24. Using this fact to change the representation of the result of the cross correlation, the magnitudes of the pixels intensities of quadrant one are switched with those of quadrant four. Similarly, quadrant two's pixels are swapped with quadrant three's. The result of the quadrant swapping as a representation of the correlation output matrix, $R(x,y)$, where the relative position offset between the template and the target is now represented by an offset of the resulting Gaussian's maximum from the center of the sequence. Figure 9 provides a gray scale representation of the result of the cross correlation, $R(x,y)$, once the quadrants are swapped. Figure 8 corresponds to the convention of defining the origin for the correlation as corresponding to the origins of the original two figures. This means that the two figures' origins are initially superimposed to create the first correlation data point. The correlation information corresponding to the four pixels which surround the center of the fields-of-view of the original figures is contained in the four corner pixels in the correlation matrix. The convention on the definition of origins attained by "swapping" is where the center of the 24×24 correlation matrix corresponds to the correlations of the centers of the two original figures.

By comparing the original Gaussian template and the noise-corrupted data with the output of the cross correlation

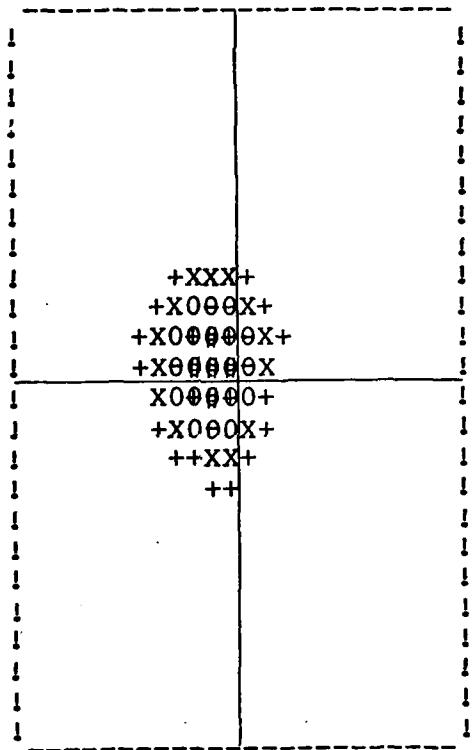


Figure 9. Diagonal Quadrant Swap of Cross Correlation

given in Figure 9, a spreading of the function is observed. The noise has also induced some asymmetries in the sequence $\underline{R}(x,y)$. The magnitudes of the components of the sequence $\underline{R}(x,y)$, are a measure of the degree of resemblance between the template and the data sequences. If the magnitude of an element of the sequence $\underline{R}(x,y)$ is less than some preset fraction of the maximum value of any element in that sequence, then that element is considered to contain poor correlation information and can be set equal to zero. The result is that such elements will have no effect on the computed offset between the template and the target. Thresholding is often done to suppress lower peaks in the result of the correlations. Figure 10 shows the result of setting this fraction to .5 for the sequence of Figure 9.

Once the threshold has smoothed the result of the cross correlation, a peak detector is usually used to find where the maximum resemblance of the two sequences occurs. Instead of a peak detector, a centroid summation was used to find the center of mass of the two-dimensional cross correlation sequence, $\underline{R}(x,y)$. The center of mass of the thresholded correlation is assumed to be good indication of the peak location. In either direction, horizontal or vertical, the centroid summation is defined by

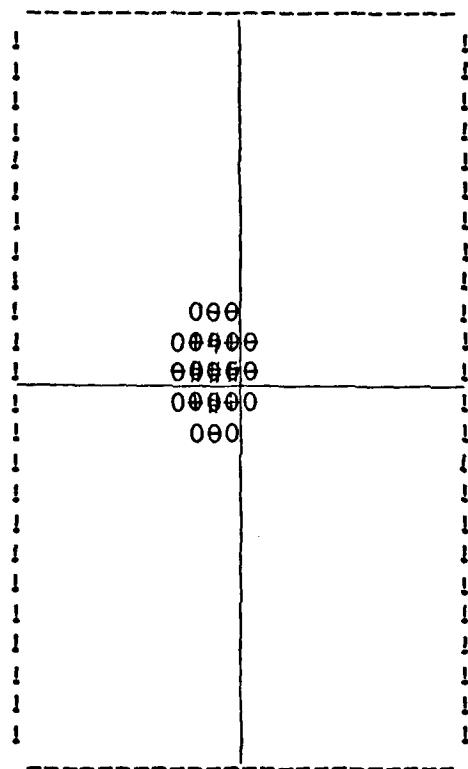


Figure 10. Result of Thresholding of .5

$$C = \frac{\sum_{i=1}^N i \cdot \text{Amp}_i}{\sum_{i=1}^N \text{Amp}_i} \quad (5-5)$$

where i is the horizontal or vertical coordinate of a given pixel, and Amp_i is the amplitude value for that pixel, and N is the total number of pixels in the array. For the horizontal direction, the horizontal coordinate's of all the elements of $R(x,y)$ would be multiplied by the amplitudes of the respective elements and the products would be summed. The resulting summation, the numerator of Equation (5-5), would then be divided by the sum of all the amplitudes. Equation (5-5) is implemented in both directions providing a position estimate of the offset of the target from the center of the data frame.

Correlator Error Statistics

This section statistically characterizes the errors in the position estimates of the correlation algorithm presented in the previous section. These position estimates will be the measurements provided to a Kalman filter which will generate a better estimate of the target's position. The two-dimensional discrete-time measurement vector, $\underline{z}_c(t_i)$, is a linear combination of the variables of interest, the target intensity function's true position, but corrupted by an uncertain measurement disturbance $\underline{v}_c(t_i)$ also of dimension two.

$$z_c(t_i) = H_c \underline{x}_c(t_i) + v_c(t_i) \quad (5-6)$$

where

$$\underline{z}_c(t_i) = \begin{bmatrix} x_{dc} \\ y_{dc} \end{bmatrix} = \text{the estimated } x \text{ and } y \text{ coordinates by the correlation/centroid algorithm}$$

$$H_c = \begin{bmatrix} 1100 \\ 0011 \end{bmatrix} = \text{the linear combination of state variables which contribute to the respective measurement elements}$$

$$\underline{x}_c = \begin{bmatrix} x_d \\ x_a \\ y_d \\ y_a \end{bmatrix} = \text{the state vector}$$

$v_c(t_i)$ = additive noise corruption, assumed to be white, and Gaussian, with statistics to be determined

The additive noise corruption vector, $v_c(t_i)$ of the measurement model of Equation (5-6), must account for the statistical effects of the errors in the correlator/centroid position estimates.

Software was developed to test the correlator/centroid algorithm's position estimates repeatedly, and histograms of the resulting error were generated. Figure 11 contains examples of the error histograms produced for a target offset by .3 pixels from the template in the horizontal direction only. The errors could be modelled as Gaussian random variables with appropriate means and variances. The values needed to describe the error's mean or variance would be a function of the template-target separation, the threshold level used by the centroid and even the variance of the

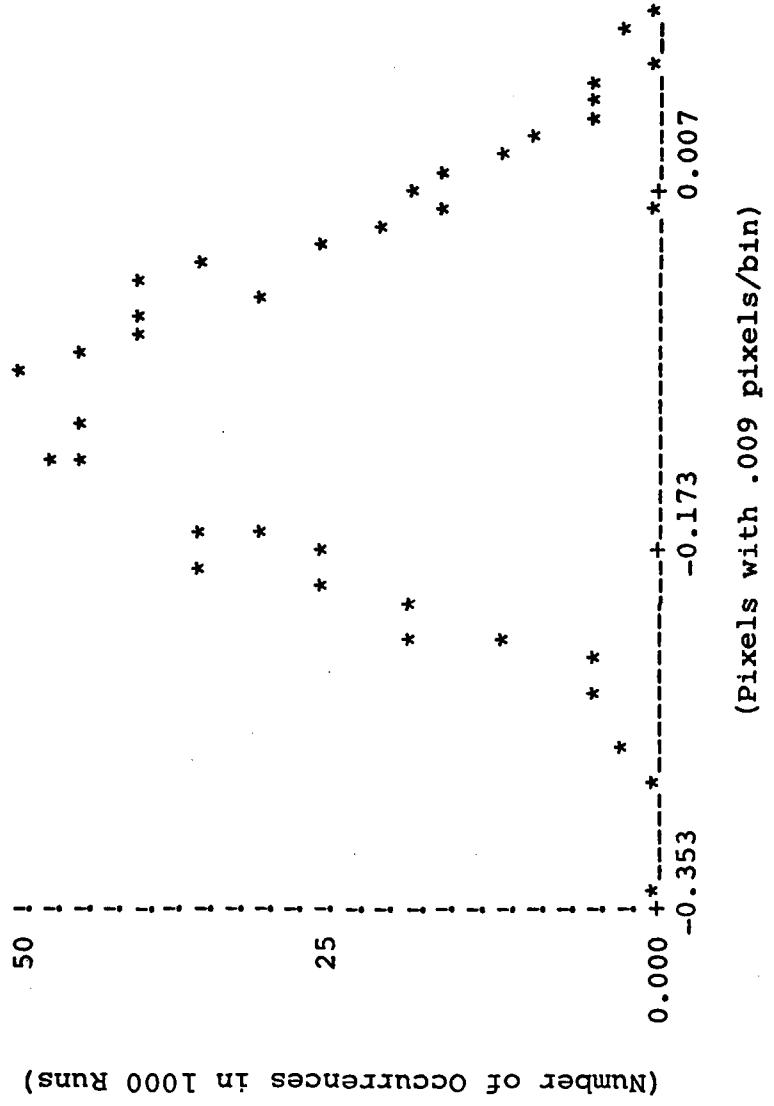
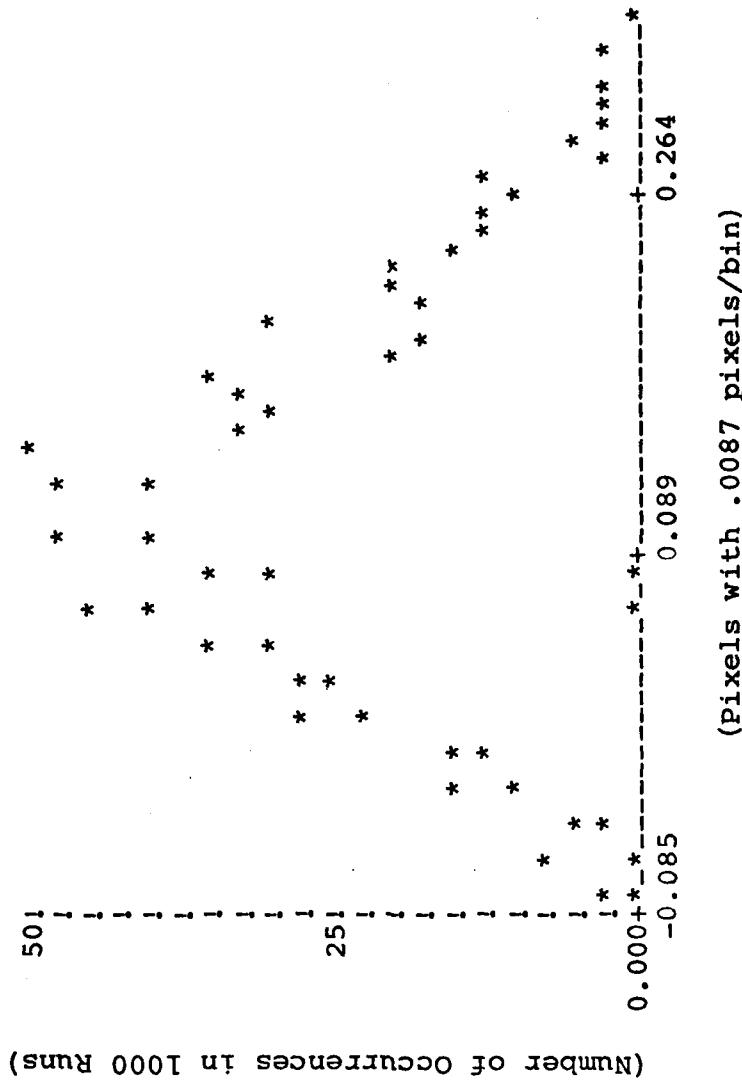


Figure 11a. Histogram of Error in Horizontal Position Estimate



background noise. For this research a value for the threshold of .5 and a target-template separation of .15 pixels were chosen to establish error statistics. Other values which were considered for the threshold were .1, .2, .25, .75, and even .90. The value of .5 was chosen since it produced acceptable results for the signal-to-noise ratios of interest. Error data for other choices of thresholds will be provided in the next chapter. At the sampling rate of 30Hz, for the trajectories being simulated in this research, a propagation error of not more than .15 pixels was expected. Background noise was allowed to vary over the full range of signal-to-noise ratios of 10 to 20. The result was a mean error of approximately -.080 pixels and a variance of .00363 pixels² in the horizontal direction. The error in the vertical direction has a mean of .116 pixel with a variance of .00598 pixels². These values can now be used to describe statistically the errors in the correlator/centroid position estimates. This characterization is needed for the measurement model of Equation (5-6).

The difference in the vertical and horizontal errors above is because of the locations of the three Gaussians of the target intensity profile. If the center of the 8 x 8 pixel tracking window is located at coordinates (0,0) then the Gaussians would be located at (0,-2.667), (-2,1.333), and (+2,1.333). This placement forces the center of mass of

the intensity profile to coordinates (0,0), but the resulting intensity pattern is not radially symmetric about (0,0).

Kalman Filter Tracker

The mathematical models for target dynamics and atmospheric jitter which were developed in Chapter IV are also used in the Kalman filter which processes the correlator's intensity function position estimates. These models accounted for time correlated dynamics and the bandwidth characteristics of the atmospheric jitter. The standard Kalman filter propagation equations, as given by Equations (4-23) through (4-26), are also used by this Kalman filter. The difference in the Kalman filter used in this alternative tracker is that it processes a two-dimensional measurement vector of intensity function position estimates, whereas the previous filter processed 64 intensity measurements. The measurement equation for the new filter was given in Equation (5-6). The low dimensionality of the measurement vector eliminates the need and in fact the applicability of the inverse covariance form of the Kalman filter update equations. The correlator/centroid intensity function's position estimates, which constitute the two-dimensional measurement vector, are linear combinations of the Kalman filter states, as seen in Equation (5-6). The linear filter measurement update equations used are given in Equations (5-7) through (5-9).

$$\underline{K}(t_i^-) = \underline{P}(t_i^-) \underline{H}^T(t_i) [\underline{H}(t_i) \underline{P}(t_i^-) \underline{H}^T(t_i) + \underline{R}(t_i)]^{-1} \quad (5-7)$$

$$\hat{\underline{x}}(t_i^+) = \hat{\underline{x}}(t_i^-) + \underline{K}(t_i) [\underline{z}(t_i) - \underline{H}(t_i) \hat{\underline{x}}(t_i^-)] \quad (5-8)$$

$$\underline{P}(t_i^+) = \underline{P}(t_i^-) - \underline{K}(t_i) \underline{H}(t_i) \underline{P}(t_i^-) \quad (5-9)$$

The values of $\underline{P}(t_i^-)$ and $\hat{\underline{x}}(t_i^-)$ are obtained from the propagation equations and the measurement uncertainty covariance matrix, $\underline{R}(t_i)$, is assumed diagonal with the values determined as explained in the previous section. The diagonal nature of $\underline{R}(t_i)$ assumes that the correlation position estimate uncertainty in one direction is independent of the uncertainty in the position estimate in the other direction. Since the errors are a function of SNR and threshold level, this independence assumption is subject to later revision. Direct computation of the off-diagonal elements of $\underline{R}(t_i)$ could verify or invalidate the assumption of independence of the errors in the position estimates.

In summary, this chapter developed an alternate tracking algorithm which uses a correlation algorithm to provide intensity function position estimates as a measurement vector to a Kalman filter. This algorithm is different from standard correlation trackers in that it uses the model of target dynamics from the Kalman filter to position the template, thresholding to remove false peaks, on-line derived template shape, and correlator position estimate enhancement via a Kalman filter. The Kalman filter uses internal models to separate what

translational motion of the intensity function resulted from target dynamics and what resulted from atmospheric jitter. The results from this tracking algorithm and the algorithm of Figure 1 are given in the next chapter.

VI. Performance Analysis

Introduction

This chapter presents the results from the testing of the tracking algorithms developed in the previous chapters. The first section of this chapter derives figures of merit which provide a means to evaluate the accuracy in the derivation of the intensity functions contained in Chapter II. This accuracy is ultimately only important in its impact on tracking ability. The sensitivity of tracking performance to the accuracy of the derived intensity functions will be shown as well. The tracking ability performance criteria are presented in the second section of this chapter. The third section discusses those variable parameters which the computer simulation allows the input to change for sensitivity analysis. The errors in the derivation of the intensity functions and the tracking errors are plotted as mean \pm one sigma (standard deviation) errors in the next section. The final section of this chapter condenses the results of this research into tables. These tables cross reference variations in the parameters which control the Kalman filter and the pattern recognition process to tracking and intensity function derivation errors. Important trends and sensitivities are revealed in these tables. The statistical information of this chapter was generated using Monte Carlo techniques (13:329), see Appendix G.

Derivation of Intensity Functions

The accuracy of the on-line derivation of the intensity functions (see Chapter II) is actually of importance here only since it affects tracking ability. To make some reasonable judgement on how good a representation of $\underline{h}(\hat{x}(t_i), t_i)$ and $\underline{H}(\hat{x}(t_i), t_i)$ has been derived, a simple figure of merit is developed in this section.

At each frame the truth model provides information on the location of the centroid of the true target's intensity functions. The data processing of Chapter II derives on-line estimates of these intensity functions. At each frame the difference between the true target intensity functions:

1. the nonlinear intensity function, \underline{h}_t
2. the derivative of \underline{h}_t with respect to a change in either horizontal state, x_d or x_a , \underline{H}_{xt}
3. the derivative of \underline{h}_t with respect to a change in either vertical state, y_d or y_a , \underline{H}_{yt}

and the estimated intensity functions, \hat{h} , $\hat{\underline{H}}_x$, and $\hat{\underline{H}}_y$, respectively, are computed pixel by pixel.

As the simulation progresses, at each frame i two 8×8 arrays are maintained to compute the accuracy of \hat{h}_i . One array is the difference between \underline{h}_{ti} and \hat{h}_i . The square of this error is also maintained: $(\underline{h}_{ti} - \hat{h}_i)^2$ where i represents which frame is being processed.

Many time histories are processed in the Monte Carlo study (as described in detail in Appendix G), and for each

frame the mean and variance of the error associated with each pixel for that frame is desired. The mean error for a given pixel, $j = 1, \dots, 64$, for a given time frame, $i = 1, \dots, 20$, over all the time histories run, $k = 1, \dots, N$, would be given by

$$M_{e_{ij}} = \frac{1}{N} \sum_{k=1}^N (h_{t_{ijk}} - \hat{h}_{ijk}) = \frac{1}{N} \sum_{k=1}^N e_{ijk} \quad (6-1)$$

where

$M_{e_{ij}}$ = mean error for pixel j at the time frame i , averaged over the N simulations.

N = number of simulations

k = index of Monte Carlo simulation runs, going from 1 to N

$h_{t_{ijk}}$ = value for frame i , pixel j , of the truth model intensity for the k th simulation

\hat{h}_{ijk} = estimate value for frame i , pixel j , of the intensity as derived in the k th simulation

The variance of the error for frame i , pixel j , would be given by

$$\begin{aligned} \sigma_{e_{ij}}^2 &= \frac{1}{N} \sum_{k=1}^N (e_{ijk} - M_{e_{ij}})^2 \\ &= \frac{1}{N} \sum_{k=1}^N (e_{ijk}^2 - 2M_{e_{ij}} \cdot e_{ijk} + M_{e_{ij}}^2) \\ &= \frac{1}{N} \sum_{k=1}^N e_{ijk}^2 - \frac{2M_{e_{ij}}}{N} \cdot \sum_{k=1}^N e_{ijk} + \frac{1}{N} \sum_{k=1}^N M_{e_{ij}}^2 \\ &= \frac{1}{N} \sum_{k=1}^N e_{ijk}^2 - 2M_{e_{ij}} \cdot M_{e_{ij}} + \frac{1}{N} \cdot N \cdot M_{e_{ij}}^2 \\ &= \frac{1}{N} \sum_{k=1}^N e_{ijk}^2 - M_{e_{ij}}^2 \end{aligned} \quad (6-2)$$

Note that in general $1/(N-1)$ could be used in place of $1/N$ in the first line of this equality to reduce the bias in this estimate, but with $N=20$ this substitution would not make much difference. Equations (6-1) and (6-2) show that two 8×8 arrays must be maintained for each frame for each intensity function of interest. In this research, the errors in \hat{h} , \hat{H}_x , and \hat{H}_y are of interest, and so six arrays were maintained for each frame.

For this research, new data frames are generated at a 30 Hz rate. If, as in Appendix G, twenty frames constitute a single tracking time history, two-thirds of a second simulation time, then six $8 \times 8 \times 20$ arrays are needed. At the end of the simulation these six $8 \times 8 \times 20$ arrays contain the information needed to compute the ensemble average and variance of errors for any pixel, j , for any frame, i . With the assumption that every pixel is as important as every other pixel, the spatial average of the mean pixel errors can be computed. Equation (6-3) shows how to compute the spatial average of the mean pixel errors for any frame, i .

$$M_{e_i} = \frac{1}{64} \sum_{j=1}^{64} M_{e_{ij}} \quad (6-3)$$

where

M_{e_i} = the spatial average (over all the pixels) of the mean pixel errors at the i th frame

$M_{e_{ij}}$ = mean (i.e. ensemble average over all the simulations) at the j th pixel for the i th frame

Note that if each pixel were not considered equally important, a weighted average could replace (6-3). Similarly, the spatial average of the variance for any frame i could be computed by

$$\sigma_{e_i}^2 = \frac{1}{64} \sum_{j=1}^{64} \sigma_{e_{ij}}^2 \quad (6-4)$$

where

$\sigma_{e_i}^2$ = the spatial average of all the ensemble pixel error variances, over all the pixels at the i th frame

$\sigma_{e_{ij}}^2$ = variance of the error of the j th pixel at the i th frame

The benchmark for intensity function derivation errors is found by providing perfect information of \hat{x}_{peak} and \hat{y}_{peak} to the pattern recognition algorithm. This benchmark is displayed as the twelfth entry of Table I in the next chapter.

Tracking Ability

Appendix G describes the Monte Carlo study used to generate the statistics of this section. The quantities of interest, with respect to tracking, are the errors in the estimated value of $\hat{x}_d(t_i^-)$, $\hat{y}_d(t_i^-)$, $\hat{x}_d(t_i^+)$, and $\hat{y}_d(t_i^+)$. It is also important to estimate the location of the intensity distribution's centroid. The accuracy of the estimation of the centroid's location, $\hat{x}_{\text{peak}}(t_{i+1}^-)$ and $\hat{y}_{\text{peak}}(t_{i+1}^-)$ of Equation (3-30a), will affect the accuracy of the estimated intensity functions and in-turn affect the tracking accuracy.

By comparing the errors before and after incorporation of a measurement, information about how the estimates were improved from the information of the data frame is obtained.

Figure G-1 shows how the error samples are generated. Similar to what was done in the previous section, error statistics are calculated as the mean error for the variable of interest at frame i,

$$\bar{E}_{x_{d_i}} = \frac{1}{N} \sum_{k=1}^N (x_{t_{d_{ik}}} - \hat{x}_{d_{ik}}) = \frac{1}{N} \sum_{k=1}^N e_{x_{d_{ik}}} \quad (6-5)$$

where

$\bar{E}_{x_{d_i}}$ = mean error (i.e. ensemble average error over all simulations) in x dynamics for frame i

$x_{t_{d_{ik}}}$ = truth model, x-dynamics value at frame i for simulation k

$\hat{x}_{d_{ik}}$ = filter estimated x-dynamic value at frame i for simulation k

and the variance of the errors,

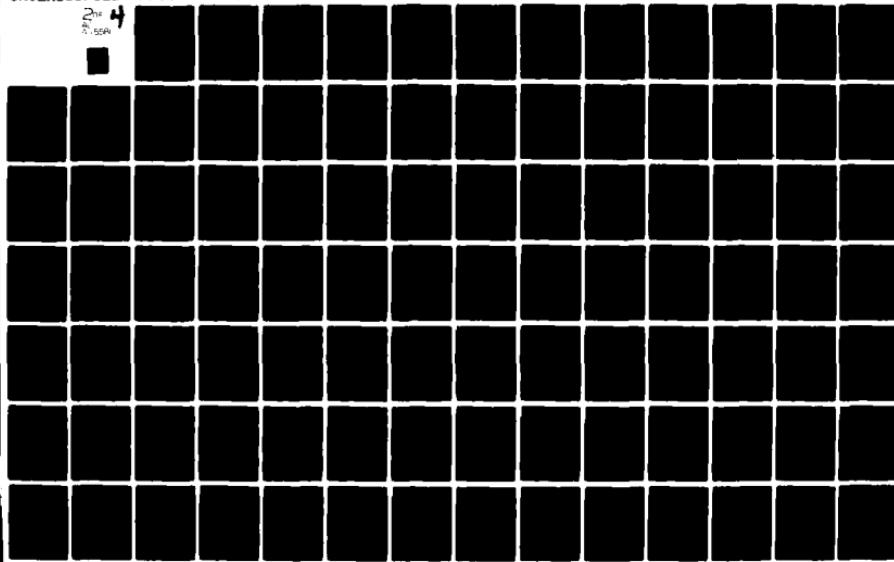
$$\sigma_{e_{x_{d_i}}}^2 = \frac{1}{N} \sum_{k=1}^N e_{x_{d_{ik}}}^2 - \bar{E}_{x_{d_i}}^2 \quad (6-6)$$

The $1/(N-1)$ substitution discussed in the previous section is also applicable here. The generalization of equations (6-5) and (6-6) to compute the errors in \hat{y}_d , \hat{x}_{peak} , or \hat{y}_{peak} is clear. The benchmark for tracking performance is set by providing perfect h , H_y , and H_x . This is the case studied by Capt Mercier (5). Table II of that research (5:57) lists a mean error of .2 pixels for a SNR = 20. That value can be

AD-A115 581 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHO0--ETC F/8 12/1
ENHANCED TRACKING OF AIRBORNE TARGETS USING FORWARD LOOKING INF--ETC(U)
DEC 81 S K ROGERS
AFIT/8E0/EE/81D-5

NL

2nd 4
2-55A



compared to entry one of Table I, contained in the next section, to show only a slight degradation in tracking ability when deriving the intensity profiles.

Variation of Parameters

The computer simulation was developed to allow certain filter and data processing parameters to be varied. This was accomplished in order to test the tracking algorithms of Figure 1 and Chapter V with different design characteristics and in different tracking environments.

The first parameter to be varied is the spread parameter, σ_g^2 , of the target's Gaussian intensity profiles. This value is used by the truth model to generate the three-Gaussian hot spot data each having the given spread, (see Equation (3-35)). The standard value for this parameter, for this research, was 2.0 but runs were also made with values of 1.0 and 3.0.

The next parameter is the number of zeros to pad around the data. Padding of the finite data array with zeros is a common engineering practice to allow manipulation of the transformed data array to provide results compatible with the limited field-of-view. In Chapter II it was shown that DFTs assume that the finite data array is one period of an infinitely periodic two-dimensional sequence. Any manipulation of the transformed data array uses this assumption of infinite periodicity. Padding with 8 rows and 8 columns of

zeros creates a 24 x 24 pixel array which is assumed to be one period by the DFT. The padding insures that the infinite periodicity assumption will not affect results within the 8 x 8 tracking window. If the spread parameter, σ_g^2 , is chosen such that the target intensity height is approaching zero near the edge of the 8 x 8 tracking window, then padding with zeros will not adversely affect the results of the data processing of Figure 1. However, if σ_g^2 is such that significant intensity magnitudes exist outside the 8 x 8 array, to pad with zeros arbitrarily would induce an artificial edge in the intensity function. These edges in the two-dimensional spatial intensity array will cause increased magnitudes of the high frequency components in the transform domain. For this application, a full frame of FLIR data actually consists of 300 x 400 pixels, so, when necessary, the 8 x 8 array could be padded with the noise-corrupted data instead of zeros. When this parameter is set to eight, then the 8 x 8 tracking window is surrounded by 8 rows and 8 columns of zeros to fill up a 24 x 24 complex data array. The standard value, for this research, for this parameter is zero causing the 8 x 8 tracking window to be padded by noise-corrupted data.

The next two parameters are the number of frames per simulation and the number of simulations per Monte Carlo study. Appendix G explains these parameters in terms of the Monte Carlo study. Both of these parameters were set to 20.

Early in the testing of the tracking algorithms, fifty frames were used to insure steady-state error was reached in a 20 frame time history. The consistency of the computed statistics, between twenty and fifty frame time histories motivated the use of the twenty frame simulation.

Alpha, the relative weighting parameter for the smoothing process, is the next parameter which may be varied. Chapter II explains how alpha affects the smoothing process, and Equation (2-9) explicitly displays its role in the algorithm. For this research, the standard value for alpha is .1, but a value for alpha of 1. and .2 are also investigated. Using alpha equal to 1. results in no smoothing and thus provides a benchmark to demonstrate what benefit smoothing provides.

The number of high frequency components of the FFT of the image to zero out is the next parameter. This provides the ability to investigate how spatial filtering within the intensity function derivation portion of Figure 1 could enhance or corrupt tracking performance. Spatial filtering could easily be accomplished within the optical implementation discussed in the next chapter if it is shown beneficial here. For standard runs, no frequencies were zeroed. Runs zeroing the two and four highest spatial frequency components were accomplished also.

The input background noise variance, σ_b^2 , is next.

This value also includes FLIR noise contributions. Since the maximum values of the three Gaussian intensities were twenty, the signal to noise ratio, SNR, is equal to

$$\text{SNR} = \frac{\text{peak signal value}}{\text{rms background noise}} = \frac{20}{\sigma_b^2} \quad (6-7)$$

SNR values of 20 (standard) to 10 were considered as representative of realistic tracking scenarios.

The next two parameters are filter parameters which are varied to tune the filter for optimal tracking. The variance of the dynamic discrete time noise driving the target states of the filter (see Equation 4-8) is a measure of the uncertainty of the filters dynamics model. The variance of the atmospherics for the filter (see Equation 4-8) is similarly explained. Without any in-depth tuning to optimize tracking performance, values of .1 and .001 pixels² were used respectively. The disparity in these values indicates the filter's atmospheric model is expected to be a very good representation of the atmospheric disturbance. The approximation of ignoring the high frequency pole made by the Kalman filter is not expected to affect the filter's atmospheric representation to any substantial degree based on performance analyses of previous filters that incorporated this same approximation (4;5;6). These values did establish acceptable tuning. Chapter VIII, Recommendations, discusses issues of tuning which should eventually be considered.

The next parameter is the RMS atmospheric jitter induced in pixels (see Equation 3-8). A value of .1 pixel jitter was used as a standard.

The software also allows the option of which domain the exponential smoothing algorithm is to be accomplished. Appendix C discusses this algorithm in both domains. Frequency domain smoothing is used for standard runs in this research.

Plotting Results

Plots of the errors in the algorithm's representation of the intensity functions and in tracking are presented in this section. For each setting of the parameter values of the previous section, 15 plots were generated. All of the plots are mean errors \pm one standard deviation.

The first four plots are of the errors in the estimates of the x and y location of the target. Errors at a minus time are the errors before incorporation of a measurement while errors at a plus time are after the measurement has been processed at that time. Plots five and six, of any 15 plot sequence, are the errors in x or y dynamic estimates at a minus time followed immediately by the errors at the plus time. Observing the way these plots are driven toward smaller errors at plus times relative to minus time estimates clearly shows how much information is obtained from that frame. Recall that the dynamics state estimates are needed

for the controller and errors in these estimates are the fundamental indicator of tracking performance, while centroid estimates are required for centering images in the upper processing path of Figure 1. Plots seven through twelve are the corresponding error plots for the centroid estimates.

Plots 13 through 15 are plots of the intensity function spatial average derivation errors of Equations (6-3) and (6-4).

For each of the fifteen plots, a legend is provided to show how the variable parameters of the previous section were set for a given plot. This legend consists of the $\text{COV} = \sigma_g^2$, NZ = number of zero to pad, rows and columns used, ALP = a used for smoothing, NF = number of frequencies to zero, VAB = variance of the background and FLIR noise, and SDF = sigma of the dynamics assumed by the filter. The following 15 plots, Figures 12 through 27, are provided here as an example of the sequence and results, for the case of COV=2, NZ=0, ALP=.1, NF=0, VAB=1., SDF=.1. Appendix I contains plots for four other settings of the variable parameters.

Figures 12 and 13 show the x and y errors at a minus time. The plots indicate that the steady state error for either variable is reached by the fourth frame. The sigma is approximately constant after that point also. The same trend is also seen in the x and y plus time errors, Figures 14 and 15, except that the errors are smaller as expected after incorporation of a measurement. Plots 16 and 17 show

how much information is obtained, and the corresponding reduction in errors, by incorporating a measurement. For these plots the errors at a minus time are plotted just prior to their corresponding plus time errors. It is clear from these plots that consistently good information is being obtained from the noise-corrupted measurements under the standard parameter settings. Figures 18 through 23 provide similar plots for the centroid location estimates. The same trend is evident for these plots except that the errors are smaller. This result was expected in that it is easier to find centroids than it is to separate dynamic and atmospheric contributions to intensity function movement. Plots 24 through 27 present the intensity function derivation errors. These plots clearly show that mean errors are reduced very quickly to very small values; the standard deviations are also reduced accordingly. The inadequacy of the dynamics models used by the Kalman filter resulted in the propagated state estimates being consistently biased in the same direction in these plots. Since it was the main goal of this research to develop and implement tracking algorithms which derive the target intensity functions in an on-line manner, the adequacy of the dynamics model was not emphasized. This problem is readily rectified by a dynamics model more representative of the target characteristics than a simple first order Gauss-Markov position process in each direction. Moreover, the propagation errors also provide a means to determine how

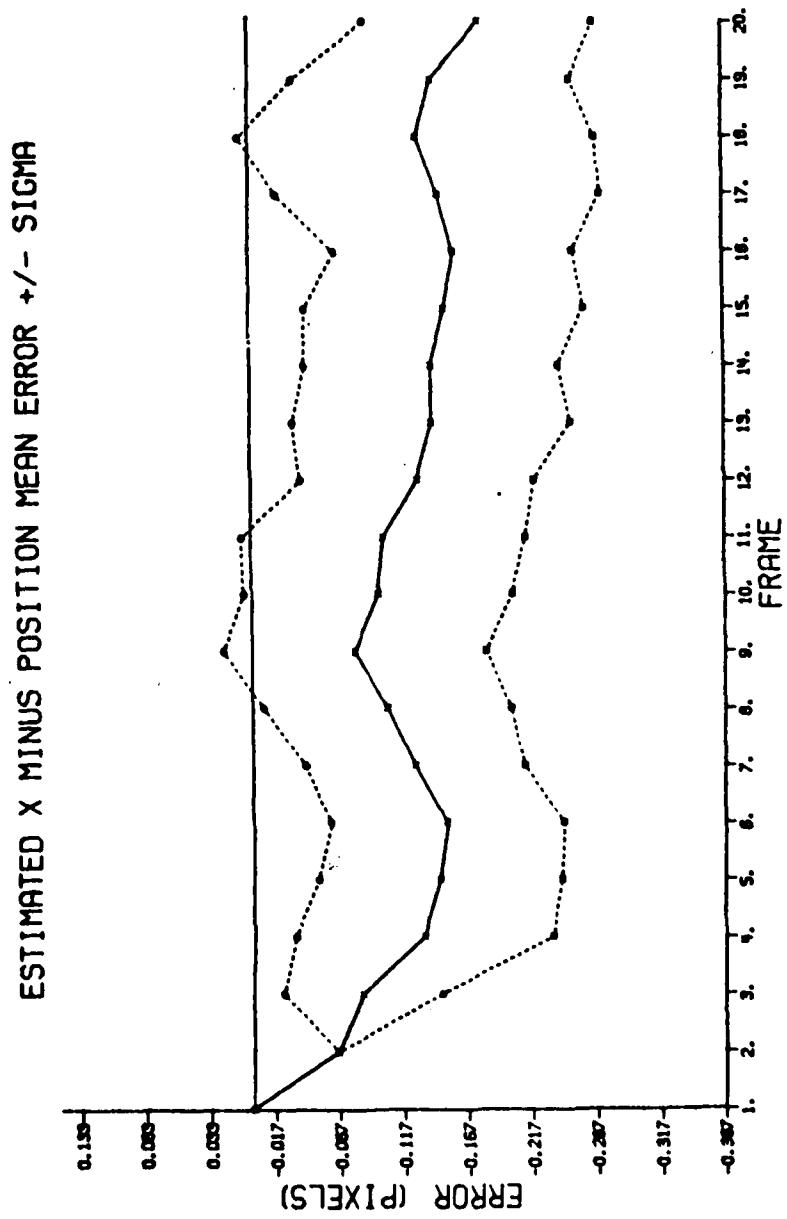


Figure 12. X Minus Errors

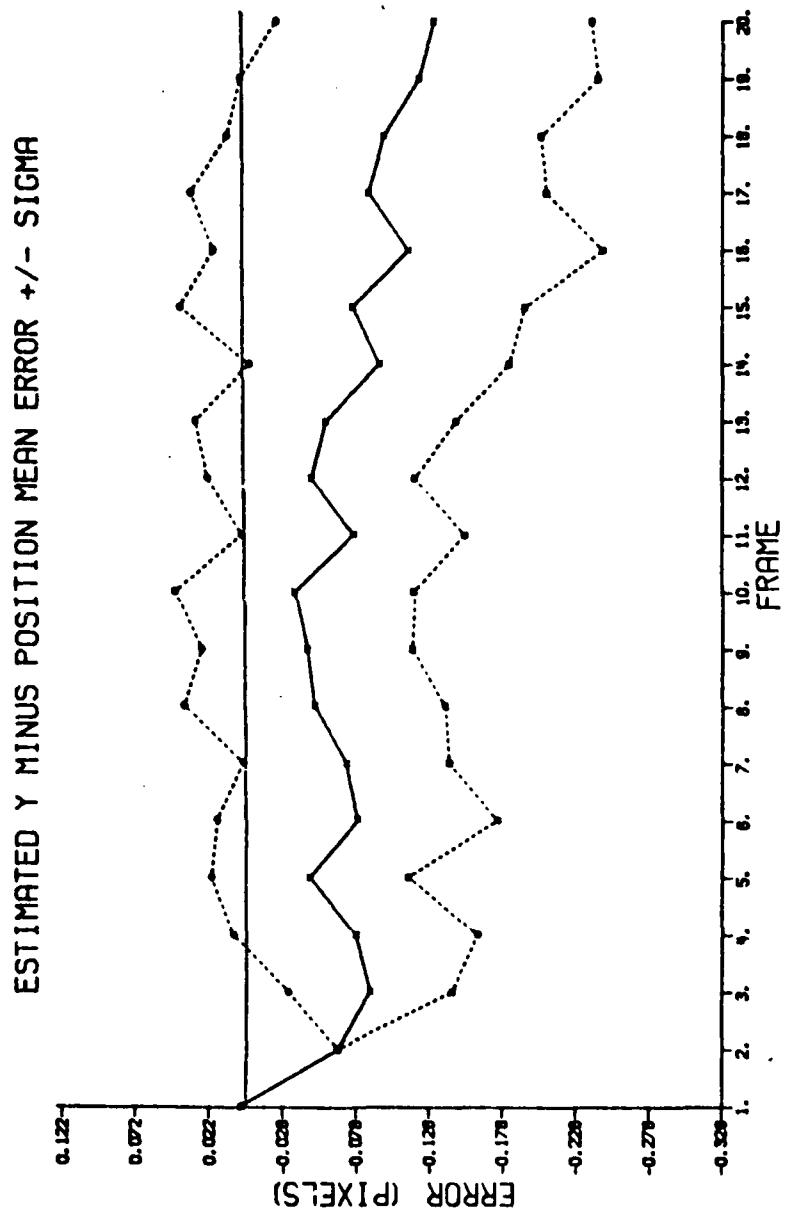


Figure 13. Y Minus Errors

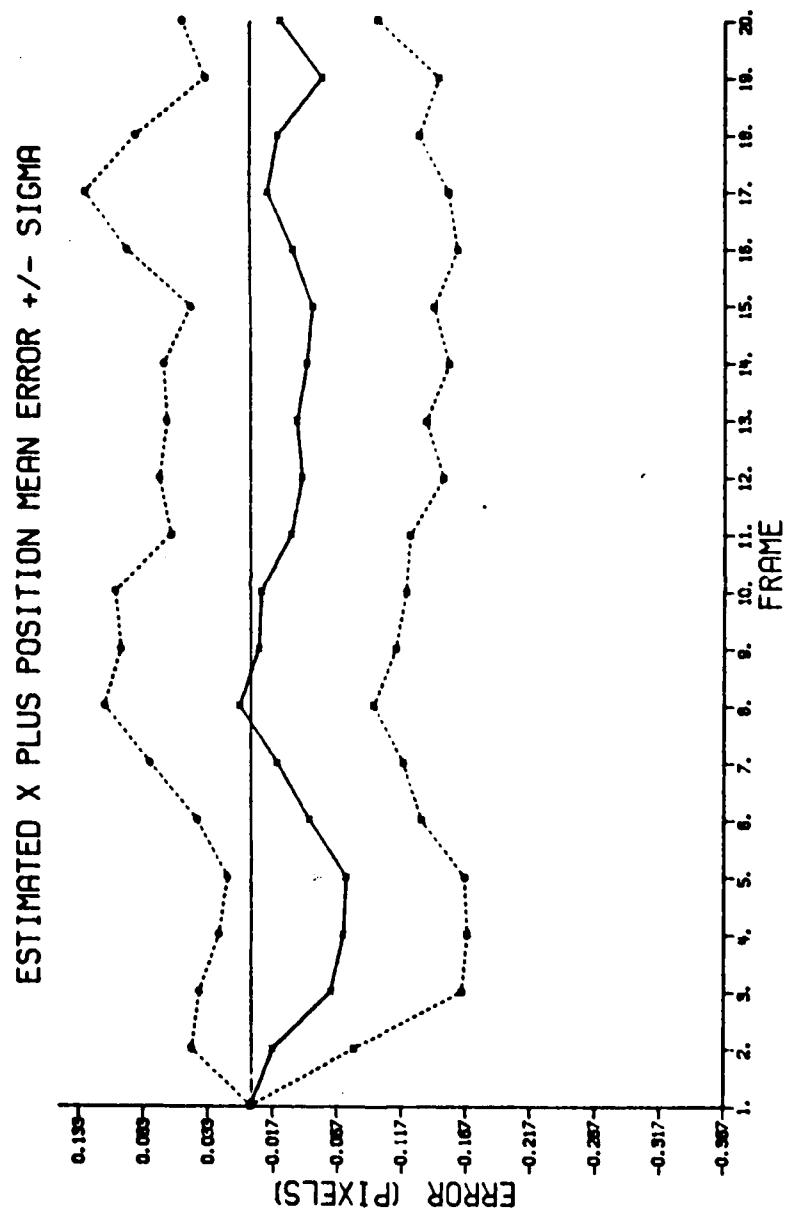


Figure 14. X Plus Error

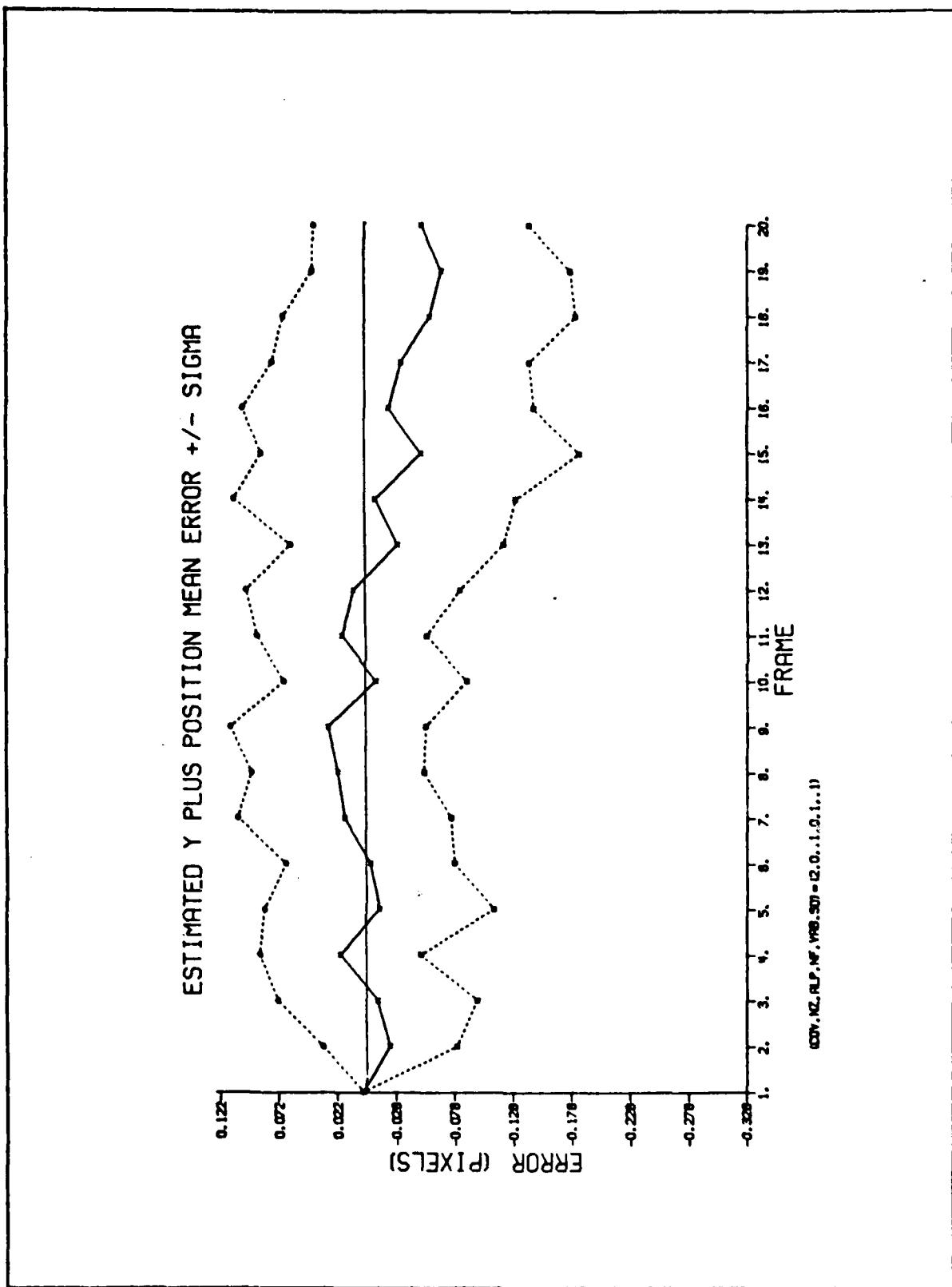


Figure 15. Y Plus Error

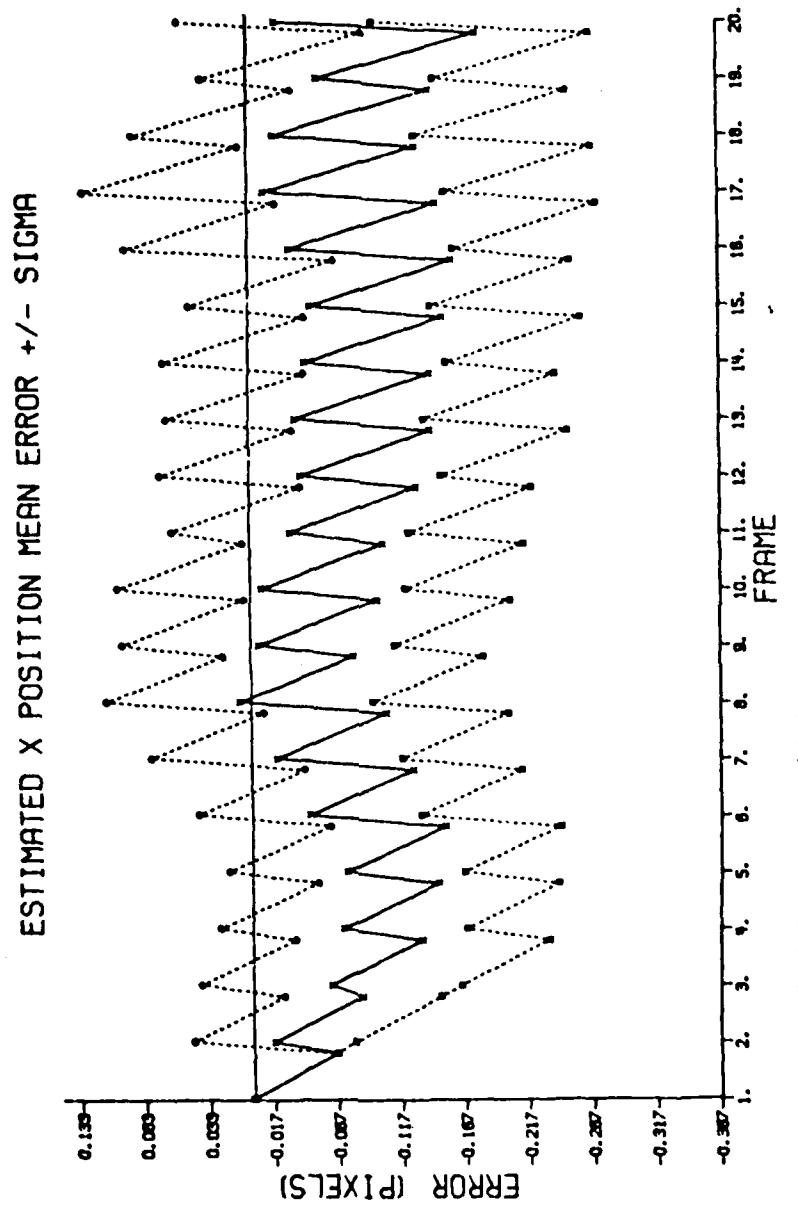


Figure 16. X Position Error: $\hat{x}_d(t_i^-)$ and $\hat{x}_d(t_i^+)$ Errors

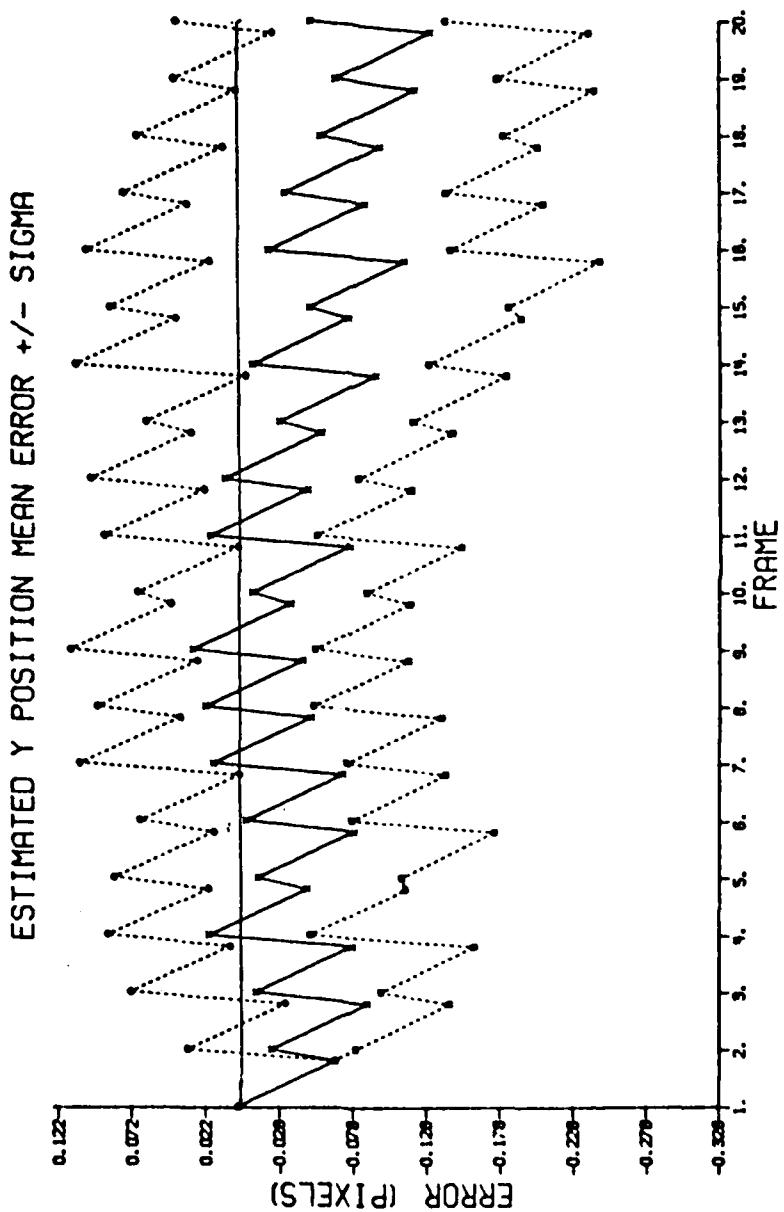


Figure 17. Y Position Error: $\hat{y}_d(t_i^-)$ and $\hat{y}_d(t_i^+)$ Errors

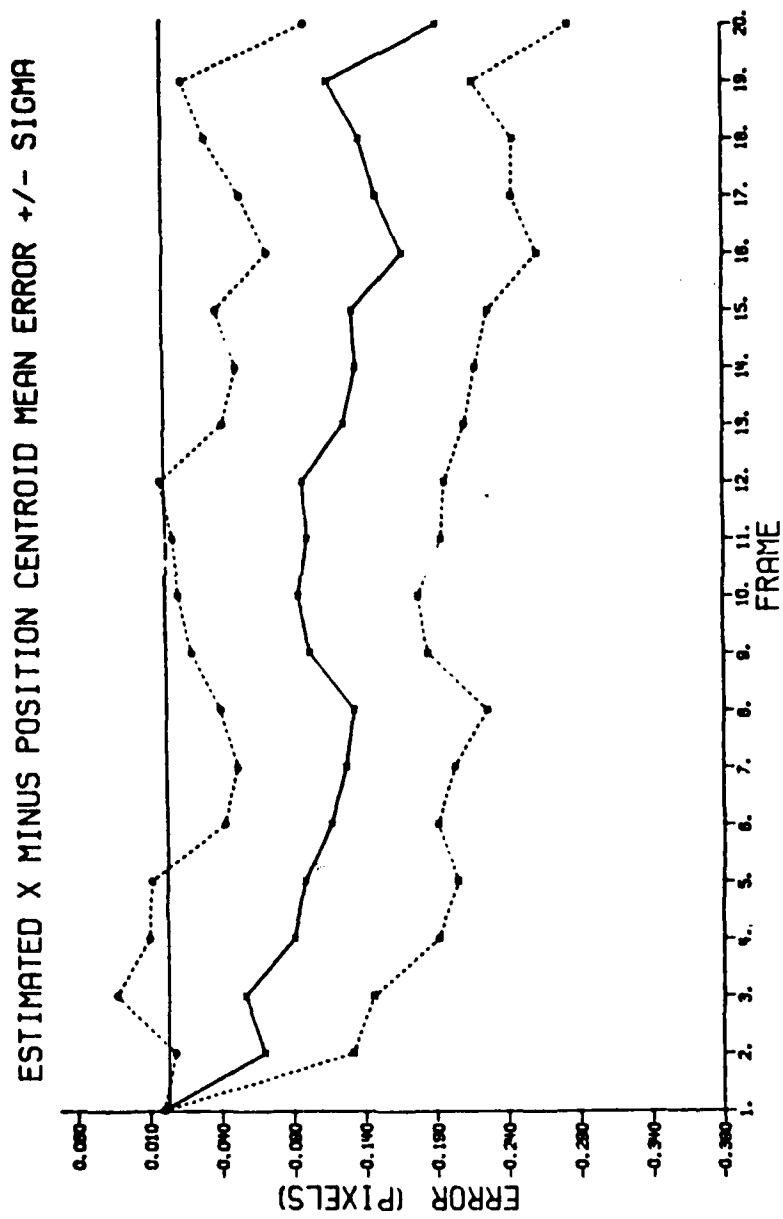


Figure 18. X Centroid Minus Error

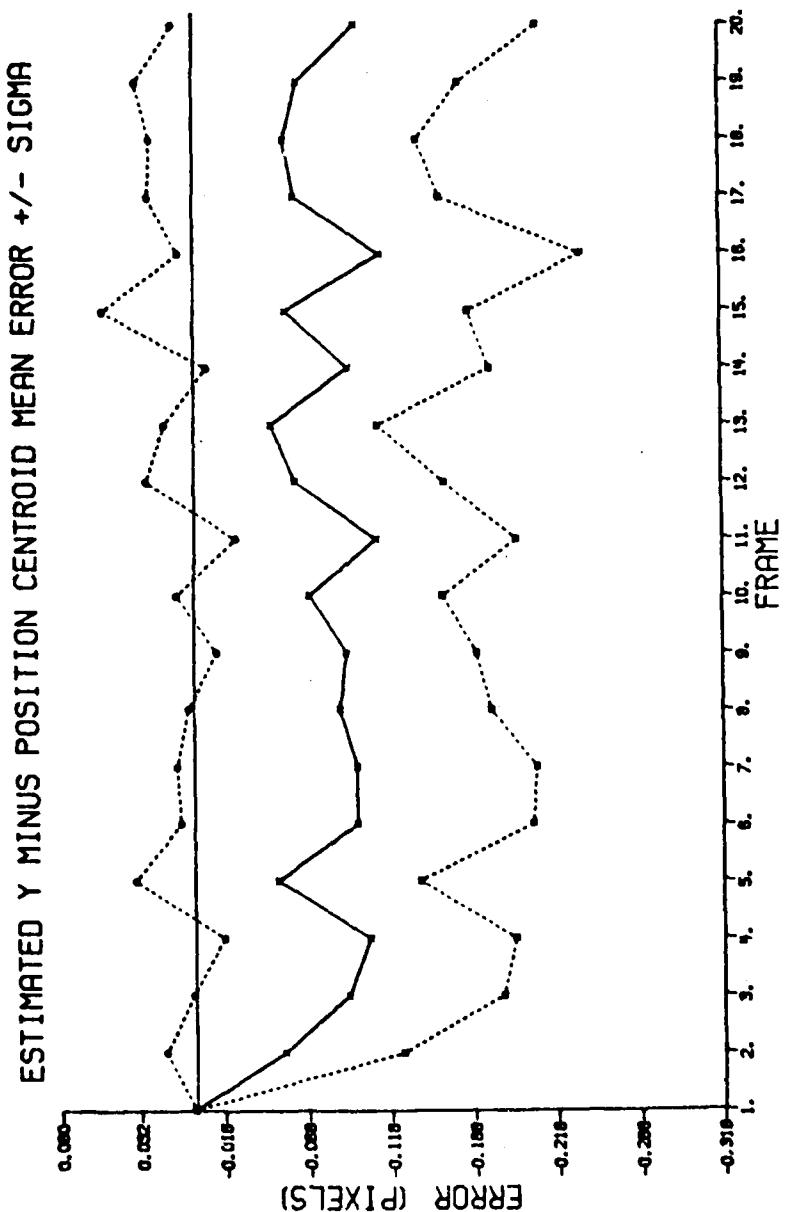


Figure 19. Y Centroid Minus Error

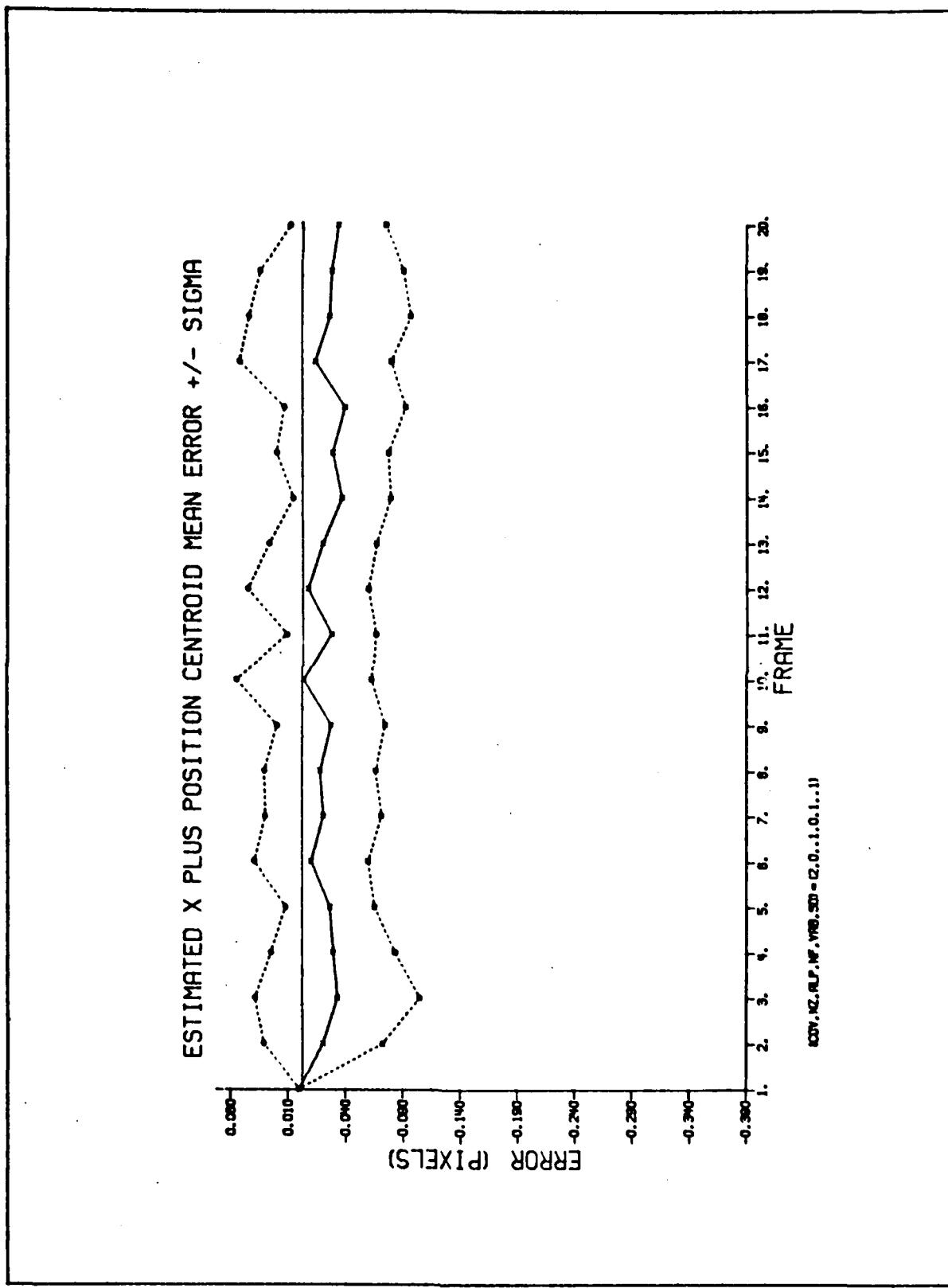


Figure 20. X Centroid Plus Error

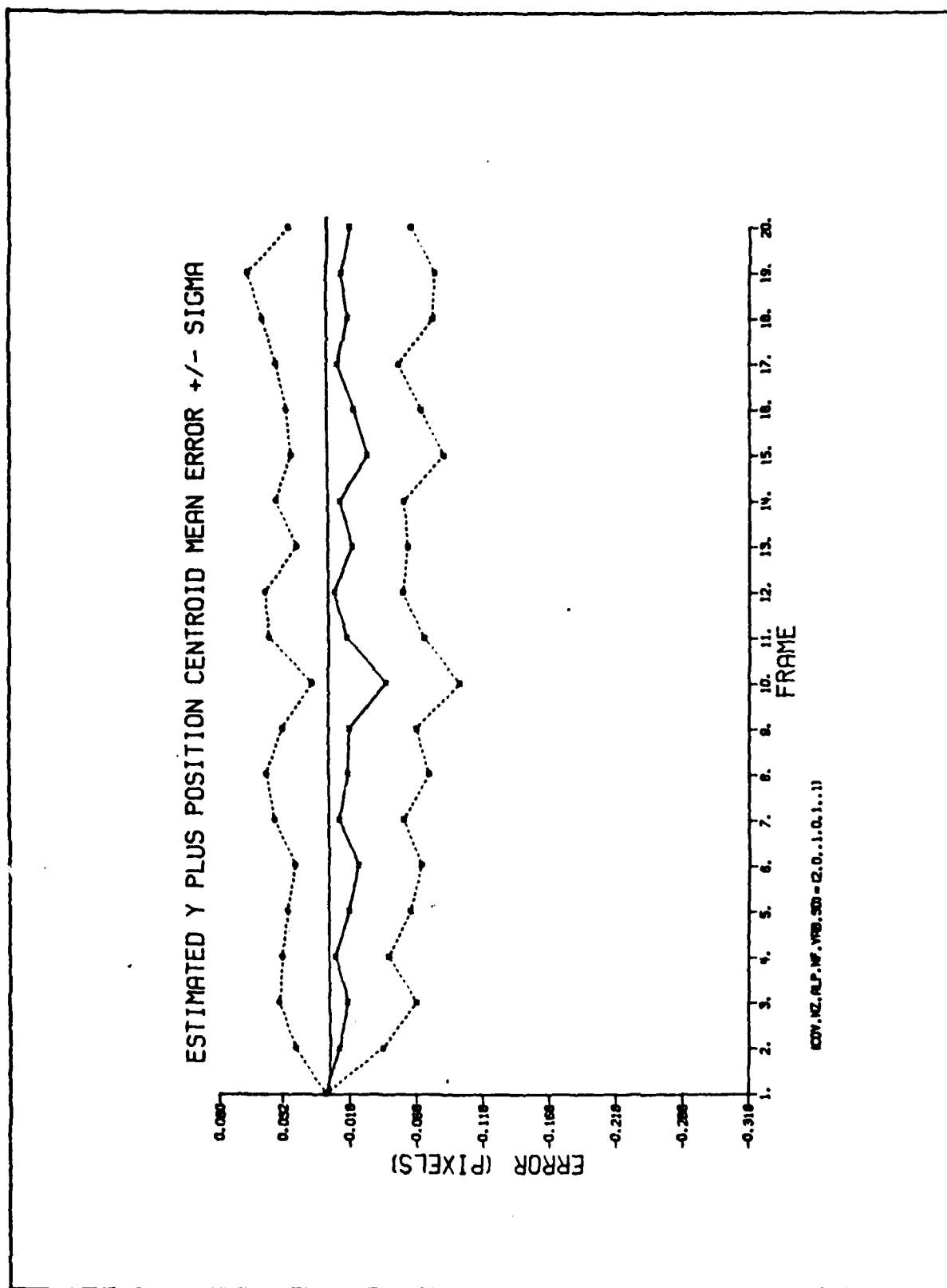


Figure 21. Y Centroid Plus Error

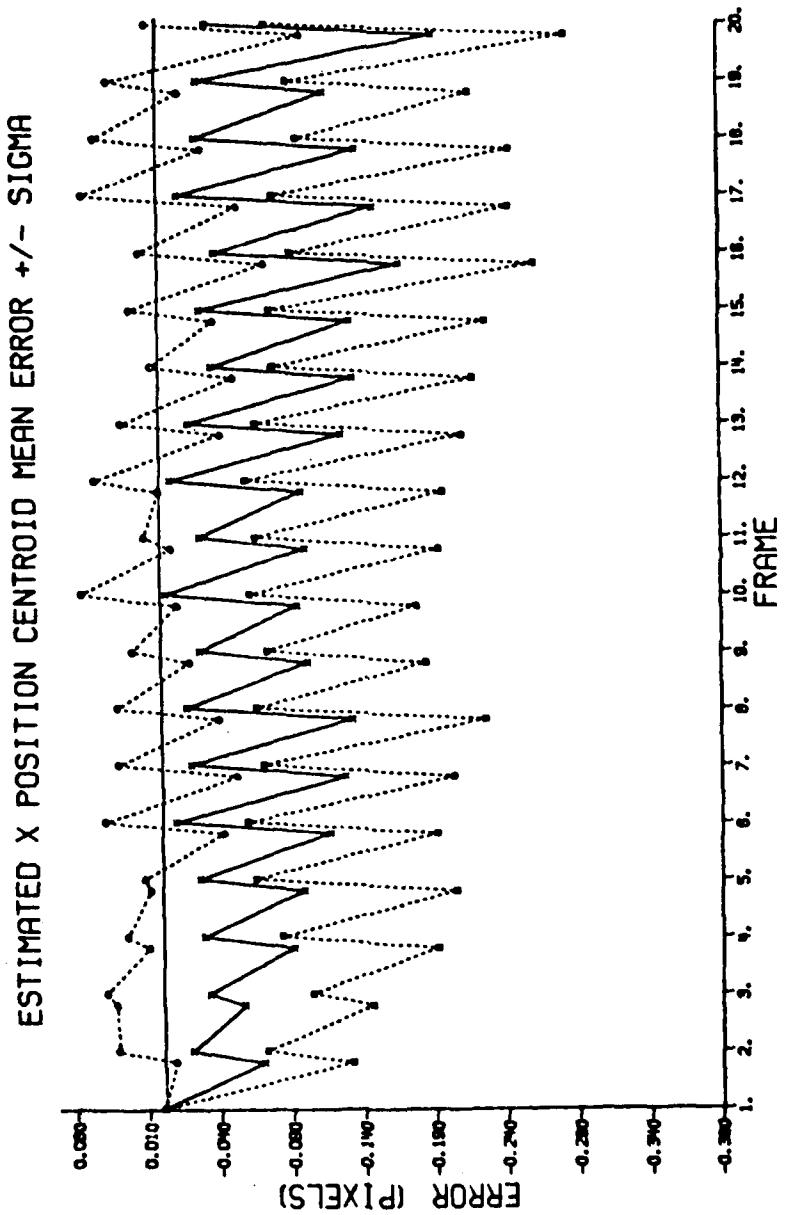


Figure 22. X Centroid Position Error: Plus and Minus Errors

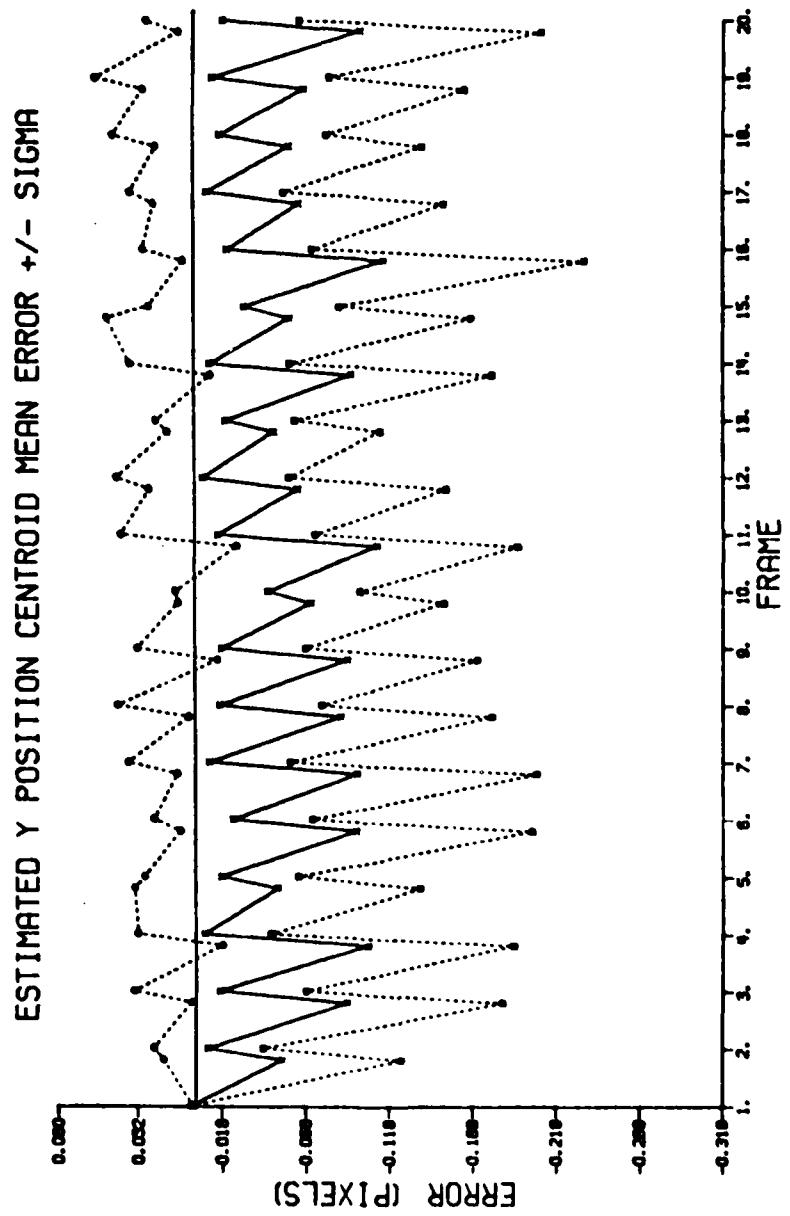


Figure 23. Y Centroid Position Error: Plus and Minus Errors

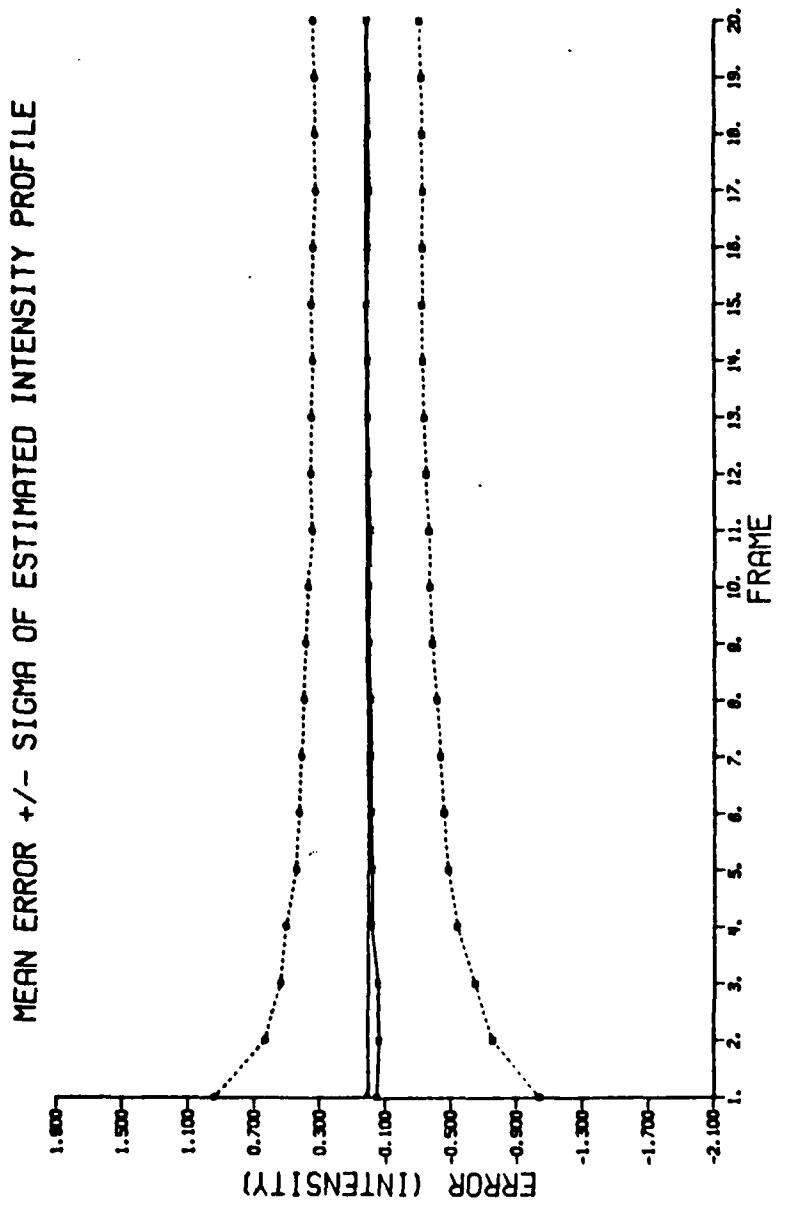


Figure 24. Error of Estimated h

MEAN ERROR +/- SIGMA OF ESTIMATED DH/DX

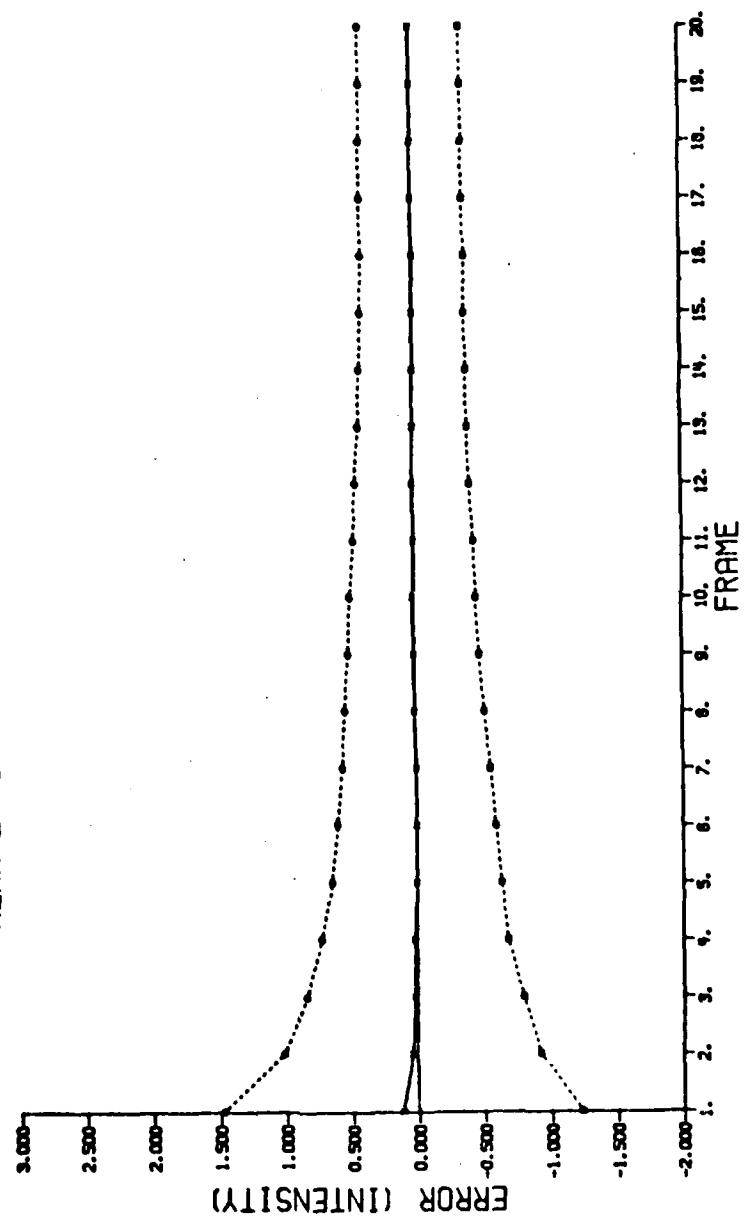


Figure 25. Error of Estimated $H_x = \frac{\partial h}{\partial x}$

MEAN ERROR +/- SIGMA OF ESTIMATED $\frac{\partial h}{\partial y}$

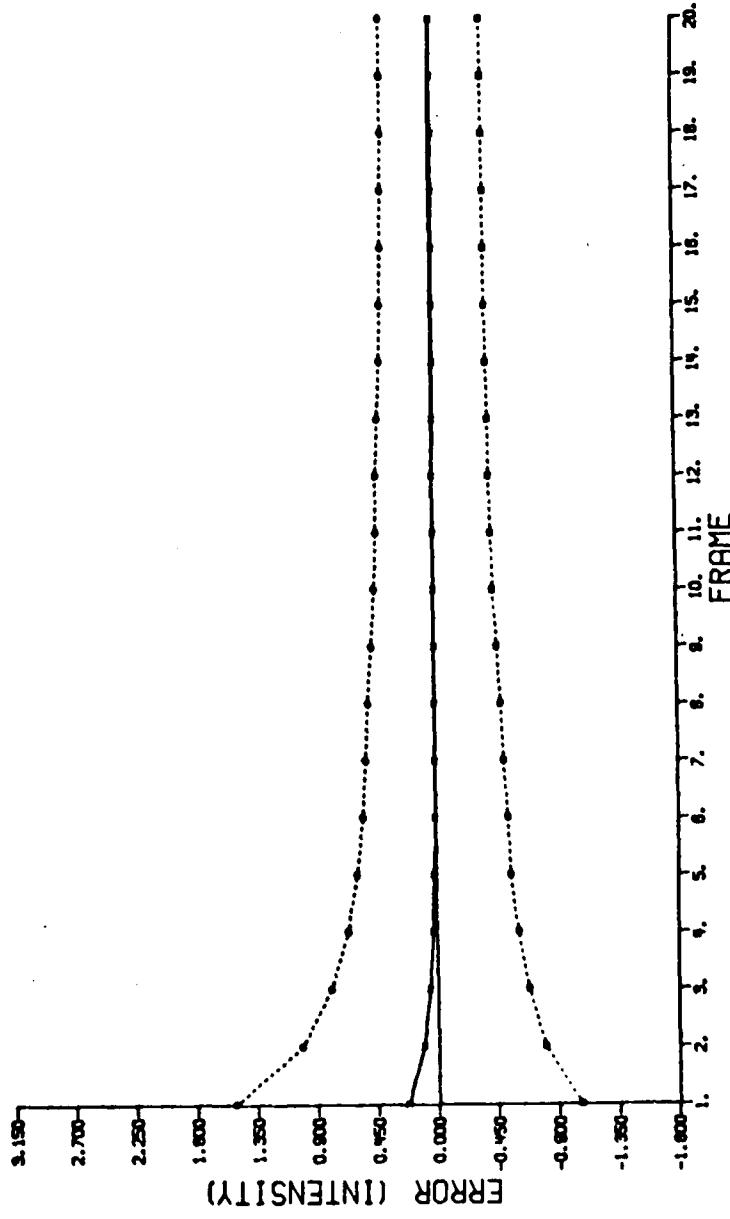


Figure 26. Error of Estimated $H_y = \frac{\partial h}{\partial y}$

much information is being obtained during measurement updates when these on-line derived intensity functions are being used.

Table Summary

This section will summarize the tracking and intensity function estimation abilities in tabular form. The plots of the previous section show errors plotted as a function of frame number. The results displayed in Tables I-III are the corresponding time average of those errors and their standard deviations over frames 10 to 20. The plots of the previous section implied that the steady state error region is reached by this time, and thus time averaging is confined to this interval. The special comment column is to designate pertinent information not indicated by the header (e.g. smooth in space, run made on Cyber 175/Modcomp minicomputer).

The first result from Table I is that exponential smoothing produces similar results in either the frequency or space domain. Entries two, four, seven, and nine of Table I are compared to the entries directly above them to show this result. This was expected from the derivation of exponential smoothing contained in Chapter II. There are additional errors induced when smoothing in space is used for this implementation. Recall Figure 1 of Chapter I. The upper path of that figure is where smoothing is accomplished. Before smoothing can be accomplished, the intensity function must be centered within the data frame. The centering is

Table I. Tracking and Intensity Function Derivation Errors
for Modcomp (minicomputer) (32 bit)

Header = (COV,NZ,ALP,NF,VAB,SDT)

Comments	Header	$\bar{X}_{\text{err}}(-)/\sigma_{\bar{X}_{\text{err}}}(-)$	$\bar{X}_{\text{err}}(+)/\sigma_{\bar{X}_{\text{err}}}(+)$	$\bar{C_n}_{\text{err}}(-)/\sigma_{C_n \text{err}}(-)$	$\bar{C_n}_{\text{err}}(+)/\sigma_{C_n \text{err}}(+)$	$\bar{h}_{\text{err}}/\sigma_{\bar{h}_{\text{err}}}$	$\bar{\partial h}/\partial x_{\text{err}}/\sigma_x \text{ err}$	$\bar{\partial h}/\partial y_{\text{err}}/\sigma_y \text{ err}$
Standard run	2,0,.1,0,1.,.1	-14237/.10685	-033996/.10559	-13567/.09443	-02635/.05220	-.00819/.11294	.05159/.16288	-.00441/.15684
Standard run Smooth in Space	2,0,.1,0,1.,.1	-14237/.10684	-03399/.10559	-13567/.09443	-02634/.05221	-.00344/.11294	.01748/.16288	-.00445/.15683
Cut 2 highest frequency	2,0,.1,2,1.,.1	-13865/.10813	-03020/.10697	-12084/.09441	-02254/.05344	-.00401/.11172	.01492/.12498	-.00096/.12681
Cut 2 freq Smooth in Space	2,0,.1,2,1.,.1	-13865/.10813	-03019/.10697	-13195/.09441	-02254/.05344	-.00412/.11172	.01491/.12697	-.00099/.12681
Cut 4 freq	2,0,.1,4,1.,.1	-13062/.10906	-02181/.10803	-12392/.09656	-01415/.05711	-.00650/.10352	.00315/.08090	-.00011/.07722
Pad zeros	2,8,.1,0,1.,.1	-15707/.10600	-05132/.10505	-15039/.09612	-04370/.05658	-.00080/.12000	.26469/.12886	.66217/.13602
Pad zeros Smooth Space	2,8,.1,0,1.,.1	-15707/.10601	-05131/.10505	-15138/.09612	-03579/.05658	-.00090/.11999	.26468/.12886	.66213/.13602
Max Noise	2,0,.1,0,4.,.1	-19813/.11769	-09482/.11970	-19151/.11787	-08732/.08694	.00031/.39491	.04328/.62031	.02009/.57596
Max Noise Smooth Space	2,0,.1,0,4.,.1	-19812/.12059	-09481/.11970	-19151/.11787	-08731/.08694	.00021/.39490	.04327/.62031	.02006/.57595

Modcomp

Header = (COV, NZ, ALP, NF, VAB, SDT)

Comments	Header	$\bar{X}_{\text{err}}(-) / \bar{\sigma}_x_{\text{err}}(-)$	$\bar{X}_{\text{err}}(+) / \bar{\sigma}_x_{\text{err}}(+)$	$\bar{C_n}_{\text{err}}(-) / \bar{\sigma}_{cn_{\text{err}}}(-)$	$\bar{C_n}_{\text{err}}(+) / \bar{\sigma}_{cn_{\text{err}}}(+)$	$\bar{h}_{\text{err}} / \bar{\rho}_h_{\text{err}}$	$\bar{\partial h} / \bar{\partial x}_{\text{err}} / \bar{\sigma}_x_{\text{err}}$	$\bar{\partial h} / \bar{\partial y}_{\text{err}} / \bar{\sigma}_y_{\text{err}}$
$\alpha = .2$	2.,0,..2,0,1,..1	-.14984/ .10516	-.13069/ .10397	-.14316/ .09382	-.03526/ .05135	.00386/ .15074	.02400/ .24566	.00548/ .23296
$\alpha = .2$ Pad zeros	2.,8,..2,0,1,..1	-.16290/ .10445	-.05827/ .10345	-.15622/ .09544	-.05067/ .05561	.00490/ .16088	.27491/ .19251	.66310/ .19765
Perfect Shift Information Benchmark	2.,0,..1,0,1,..1	-	-	-	-	-.006996/ .06526	.02115/ .1536	.00314/ .13646

Table II. Tracking and Intensity Function Derivation Errors
for Cyber

Header = (COV, NZ, ALP, NF, VAB, SDT)

Comments	Header	\bar{X}_{err} (-) / $\sigma_{\bar{X}_{\text{err}}}$ (-)	\bar{X}_{err} (+) / $\sigma_{\bar{X}_{\text{err}}}$ (+)	\bar{C}_n err (-) / $\sigma_{C_n \text{err}}$ (-)	\bar{C}_n err (+) / $\sigma_{C_n \text{err}}$ (+)	\bar{h}_{err} / $\sigma_{\bar{h}_{\text{err}}}$	$\frac{\partial \bar{h}}{\partial x} \text{err}$ / $\sigma_x \text{err}$	$\frac{\partial \bar{h}}{\partial y} \text{err}$ / $\sigma_y \text{err}$
Standard run	2,0,.1,0,1.,.1	-.16401/ .09851	-.06009/ .10155	-.14510/ .09683	-.03973/ .05396	-.01362 .12596	.01171/ .16935	-.00240/ .16071
Cut 2 highest frequency	2,0,.1,2,1.,.1	-.16133/ .09892	-.05716/ .10179	-.14246/ .09684	-.03682/ .05313	-.01593/ .11595	.00765/ .12728	.00080/ .12772
Cut 4 highest frequency	2,0,.1,4,1.,.1	-.15310/ .09985	-.04853/ .10316	-.13398/ .09763	-.02795/ .05532	-.01875/ .10966	-.00285/ .08395	.00526/ .08908
Pad zeros	2,8,.1,0,1.,.1	-.18205/ .08699	-.08080/ .09138	-.16393/ .09083	-.06079/ .05181	-.01355/ .14139	.25773/ .14166	.65931/ .15111
Pad zeros Cut 2 freq	2,8,.1,2,1.,.1	-.15440/ .08918	-.05040/ .09328	-.13539/ .09126	-.029931 .04822	-.05099/ .10413	.27265/ .10348	.34126/ .10350
Pad zeros Cut 4 freq	2,8,.1,4,1.,.1	-.15217/ .08814	-.04815/ .09227	-.13309/ .09116	-.02762 .04804	-.09408/ .08726	.26509/ .06571	.28585/ .05512
Max Noise	2,0,.1,0,4.,.1	-.25885/ .12018	-.15837/ .12461	-.24168/ .12601	-.14068/ .10081	-.01382/ .48062	.04630/ .105467	.05107/ .95636
Max Noise Cut 2 freq	2,0,.1,2,4.,.1	-.25601/ .11867	-.15474/ .12271	-.23904/ .09363	-.13636/ .09363	-.01692/ .45704	.03043/ .78877	.03624/ .72683
Max Noise Cut 4 freq	2,0,.1,4,4.,.1	-.23200/ .12373	-.12861/ .12980	-.10967/ .10119	-.02014/ .46506	-.00930/ .47468	.03069/ .45459	

Cyber

Header = (COV,NZ,ALP,NF,VAB,SDT)

	$\bar{X}_{\text{err}}(-) / \sigma_{\bar{x}_{\text{err}}}(-)$	$\bar{X}_{\text{err}}(+) / \sigma_{\bar{x}_{\text{err}}}(+)$	$\bar{C_n}_{\text{err}}(-) / \sigma_{C_n \text{ err}}(-)$	$\bar{C_n}_{\text{err}}(+) / \sigma_{C_n \text{ err}}(+)$	$\bar{h}_{\text{err}} / \sigma_{h_{\text{err}}}$	$\bar{\partial h} / \partial \bar{x}_{\text{err}} / \sigma_{\bar{x}_{\text{err}}}$	$\bar{\partial h} / \partial y_{\text{err}} / \sigma_y \text{ err}$
Set broad gaussians $\sigma_g^2=3.$	-.17166/.10268	-.06887/.10694	-.15289/.10254	-.04865/.06478	.01443/.15304	.03733/.15006	-.02182/.16940
Broad gaussian pad zeros	-.25443/.07859	-.15775/.08227	-.23708/.09335	-.13904/.06827	-.02478/.18966	.96095/.13722	1.12159/.15447
Sharply peaked gaussian $\sigma_g^2=1.$ max noise	-.20193/.10531	-.09843/.10804	-.18386/.17194	-.07891/.07194	.03004/.38217	.11787/.95185	.06475/.91628
Sharply peaked max noise Pad errors	-.18863/.10873	-.08510/.11151	-.17026/.10690	-.06527/.07131	.01977/.37866	.07142/.54349	.16721/.51137
Sharply peaked max noise Pad zeros cut 4 freq	-.16987/.11241	-.05443/.11556	-.15109/.10979	-.03619/.07453	-.01376/.29974	.01352/.31264	.10246/.26368
Sharply peaked cc=.2	-.15465/.09155	-.05060/.09321	-.13560/.08959	-.03009/.03813	.00430/.13314	.02005/.24674	-.00978/.22716

Cyber

Header = (COV, NZ, ALP, NF, VAB, SDT)

		$\bar{X}_{\text{err}}(-) / \sigma_{x_{\text{err}}}(-)$	$\bar{X}_{\text{err}}(+) / \sigma_{x_{\text{err}}}(+)$	$\bar{Cn}_{\text{err}}(-) / \sigma_{c_{\text{err}}}(-)$	$\bar{Cn}_{\text{err}}(+) / \sigma_{c_{\text{err}}}(+)$	$\bar{h}_{\text{err}} / \sigma_{h_{\text{err}}}$	$\bar{\partial h} / \partial x_{\text{err}} / \sigma_x_{\text{err}}$	$\bar{\partial h} / \partial y_{\text{err}} / \sigma_y_{\text{err}}$
$\alpha=.2$	2.,0.,2,0,1,,,1	-.17092/ .09758	-.06822/ .10061	-.15211/ .09663	-.04797/ .05433	-.01267/ .16021	.01599/ .25261	-.00094/ .22793
$\alpha=.2$ cut 4 freq	2.,0.,2,4,1,,,1	-.15695/ .09995	-.05310/ .10328	-.13789/ .09764	-.03258/ .05590	-.01798/ .13531	-.00014/ .11204	.00460/ .11618
$\alpha=.2$ max noise	2.,0.,2,0,4,,,1	-.23271/ .11564	-.18662/ .12021	-.26580/ .12513	-.16843/ .10165	-.00989/ .63206	.08033/ .1.56522	.09211/ .1.43047
No smoothing	2.,0,1.,0,1,,,1	-.45490/ .09643	-.38392/ .10125	-.44009/ .10821	-.36811/ .08998	-.03799/ .1.134	-.01214/ .1.7261	.12642/ .1.5972
No smoothing cut 4 freq	2.,0,1.,4,1,,,1	-.28138/ .10730	-.19140/ .11404	-.26407/ .10679	-.17283/ .81812	-.018096/ .83334	.004024/ .78588	.024496/ .68377

Table III. Tracking Errors for Correlator-Kalman Filter
for Modcomp

Header = (COV,NZ,ALP,NF,VAB,SDT)

Comments	Header	$\bar{X}_{\text{err}}(-)/\sigma_x \text{ err}(-)$	$\bar{X}_{\text{err}}(+)/\sigma_x \text{ err}(+)$	$\bar{C}_n \text{ err}(-)/\sigma_{C_n} \text{ err}(-)$	$\bar{C}_n \text{ err}(+)/\sigma_{C_n} \text{ err}(+)$	$\bar{h}_{\text{err}}/\sigma_h \text{ err}$	$\frac{\partial \bar{h}}{\partial x} \text{ err}/\sigma_x \text{ err}$	$\frac{\partial \bar{h}}{\partial y} \text{ err}/\sigma_y \text{ err}$
Standard run	2,0,.1,0,1,.,.1	-.13517/.13668	-.02543/.13437	-.12851/.12413	-.01783/.09694			
Pad zeros	2,8,.1,0,1,.,.1	-.20829/.12931	-.10405/.12850	-.20169/.11900	-.09655/.09655			
Cut 2 highest frequency	2,0,.1,2,1,.,.1	-.13610/.13630	-.02652/.13395	-.12945/.12440	-.01892/.09711			
Cut 4 highest frequency	2,0,.1,4,1,.,.1	-.13466/.13512	-.02513/.13285	-.12801/.12330	-.01753/.09587			
Max Noise	2,0,.1,0,4,.,.1	-.14071/.22386	-.03185/.22057	-.13408/.21936	-.02427/.20430			
Max Noise Cut 2 freq	2,0,.1,2,4,.,.1	-.13946/.22266	-.03047/.21967	-.13282/.21747	-.02288/.20269			
Max Noise Cut 4 freq	2,0,.1,4,4,.,.1	-.13929/.22369	-.03035/.22058	-.13265/.21829	-.02277/.20315			
ALPHA = .2	2,0,.2,0,1,.,.1	-.13881/.13582	-.03009/.13338	-.13217/.12330	-.02251/.09636			

accomplished in the frequency domain, (see Chapter II). If smoothing is to be accomplished in the space domain, the centered data must then be inverse transformed. The smoothed data would then be transformed again for evaluation at the propagated state estimate and differentiation which are accomplished in the frequency domain. The additional transforms required for smoothing in space induce additional numerical errors. The very slight difference between the results shows that transforming errors are practically negligible. Smoothing in space may be used for the optical implementations which will be presented in the next chapter.

Table I contains the results obtained on the Modcomp while Table II contains the results obtained on the Cyber 175. The Modcomp minicomputer uses 32 bit arithmetic compared to the Cyber's 60 bits. The first entry from either table is the standard run as defined by the parameter settings contained in the header. The comparable results obtained in tracking errors for example in $\hat{x}_{err} (+)$, $-.03399 \pm .10559$ compared to $-.06009 \pm .10155$, reveal an insensitivity of the developed algorithms to word length. Similarly, comparable results were obtained for filtering out high frequencies, padding with zeros, and even the maximum noise corruption case. This result is important since it implies an insensitivity to h , H_x , and H_y errors. The insensitivity also implies that h and the Kalman filter gain, which uses H_x and

H_y , could be refreshed at a lower cycle rate than the 30 Hz frame cycle time.

Zeroing out the two highest spatial frequencies is seen to increase accuracy on all columns in Table I. This was expected for the spread variable, σ_g^2 , equal to two. (This will be explored further when broad and sharp intensity shapes are discussed below.) The values for the three Gaussian dispersions, whose peaks are separated by approximately four pixels, results in an intensity function which contains relatively low spatial frequencies. The high spatial frequency components are because of noise and the zeroing of them decreases the intensity function derivation errors and thereby also increases tracking accuracy. Tables I and II show that zeroing out the four highest spatial frequencies didn't result in a significant change in the errors from the two frequency cut off case.

Padding with zeros slightly increases errors over the pad with data case for the standard parameter settings. This shows that the true intensity profile is not equal to zero outside the 8 x 8 tracking window for the spread parameter equal to 2. To pad with zeros introduces artificial high frequencies as explained earlier, and zeroing out these artificial high frequencies improved tracking by 18 percent for $\hat{x}(-)$ (see Table II entries 5 and 6), where only a 2 percent improvement was achieved when padding with data (see Table II

entries 1 and 2). For this application the luxury of padding with data exists because a frame of FLIR data actually consists of 300 x 400 pixels. Padding with zeros will be shown to enhance tracking for sharp intensity profile cases below. For sharp target intensity profiles the true target intensity is approximately zero outside the 8 x 8, so only noise contributes significant amplitudes there. Zeroing that noise enhances tracking.

The eighth entry of Table I provides the tracking and intensity function derivation errors when SNR is at its minimum level used in this research, i.e., SNR = 10. The increased noise did degrade tracking, by about 40% for $\hat{x}(-)$ and by about 300% for $\hat{x}(+)$, as was expected since it becomes harder to find the target within the data frame. The derivation errors for h, for the maximum noise corruption case of Table II entry seven, is seen not to change significantly from the case of high standard SNR as in entry one. This result shows that the pattern recognition algorithm is not very sensitive to the increase in noise.

Entries four, seven, and nine of Table I present information on the combination of smoothing in space with cutting the two highest spatial frequencies, padding with zeros or in combination with maximum noise corruption. Smoothing in space for all of these cases produced results compatible with smoothing in the frequency domain. This is seen when

each of the smoothing in space entries are compared with the entries directly above them in Table I.

Entry ten of Table I and entry sixteen of Table II result from using an alpha equal to .2 instead of the standard .1. Tracking ability is degraded from that of entry one even though errors in the derivation of \underline{H}_x and \underline{H}_y are not increased. However, errors in deriving \underline{H}_x do increase. Since setting alpha equal to .2 simulates a finite memory filter of about five samples long, this result shows that a higher magnitude steady state error level is reached for \underline{H}_x because less smoothing is accomplished. But, if the target shape does change, this larger alpha would respond faster than the smaller standard alpha. Recall that the placement of the three Gaussians, as explained in Chapter V, is the reason that \underline{H}_x is harder to derive than \underline{H}_y . The no smoothing entry of Table II shows that no smoothing increases tracking errors, $\hat{x}(+)$, by almost 400%. Cutting four frequencies reduces tracking errors to about 300% of the smoothed error.

The sharply peaked intensity entries of Table II show that tracking and intensity function derivation errors increase over the corresponding broad intensity cases. The increased errors are because the target's intensity functions are not as easily distinguished from the noise as when the target has a broad intensity profile. Padding with zeros for the sharply peaked intensity entries of Table II decreases

tracking errors by 10%. Recall that padding with zeros increased errors for the $\sigma_g^2 = 2$ case. The placement of the Gaussians with small dispersions result in the target only contributing insignificant magnitudes outside the 8 x 8 tracking window. The padding with zeros only essentially zeros only noise for these cases resulting in smaller errors.

Last it should be noted that the Correlator-Kalman filter algorithm of Chapter V outperformed the Figure 1 algorithm consistently. The first entry of Table III shows that for the standard case that the correlator-Kalman filter algorithm has smaller tracking errors, 25% less for $\hat{x}(+)$, and centroid location errors, 32% less for $\hat{x}_{peak}(+)$. The sensitivity of this algorithm to the variation of parameters is similar to the results discussed above. Padding with zeros increased errors, by 35% for $\hat{x}(-)$, while cutting high frequencies did not have much effect on the standard case. Maximum noise increases errors, by 20% for $\hat{x}(+)$, but cutting high frequencies from this minimum SNR case regains 9% of that lossed accuracy. Setting alpha equal to .2, entry 8 Table III, is found not to induce additional tracking errors, unlike the effect it had on the algorithm of Figure 1. Since this algorithm does not use H_x , this result was expected. Also it must be re-emphasized that this is not a standard correlator. This correlation algorithm uses dynamic modelling information from the Kalman filter to position the template.

Thresholding is used to avoid false peaks along with increasing the accuracy of the centroid calculation. Thresholds of .1 and .9 were tested for no noise and for SNRs of 10 and 20. The mean error, for SNR = 10, decreased from .08211 pixels for no threshold to -.03870 pixels for a threshold of .2. Although these mean errors are both close to zero and their difference may be considered insignificant, the standard deviation made a significant decrease from .01357 to .00531 pixels. At a threshold of .5 the mean error had increased to .080 pixels and the standard deviation had dropped to .00363 pixels. The histogram of the errors also approximated the Gaussian form more than the .2 or .3 threshold cases. This Gaussian form along with the tighter standard deviation led to the choice of using a threshold of .5 for this research. The third enhancement of the correlator is that the template is being derived on-line via the data processing of Chapter II. Finally, the enhanced correlator position estimates are processed by a Kalman filter. Capt Mercier's research (5) tested a standard correlation tracker against an extended Kalman filter algorithm which utilized apriori knowledge of the intensity functions. In Table II of that study (5:57) the mean track error, for a standard correlation tracker, is listed at .5 pixels with a 1σ error of 1.5 pixels for a SNR = 20. The correlation algorithm developed for this research resulted in a mean error of -.02543 pixels with a 1σ

error of .13437 pixels. This algorithm shows considerable potential as a next generation tracker.

Summary

This chapter presented the results from the testing of the tracking algorithms developed in the previous chapters. The errors are plotted as mean \pm one sigma (standard deviation) errors and tables of errors are generated for variable settings for Kalman filter and pattern recognition parameters. Extremely good tracking is established for both algorithms with the Correlator-Kalman filter algorithm showing considerable potential.

The better performance of the alternate tracking algorithm can be explained as follows. Previous comparisons (4;5) of Kalman filter trackers to correlators used apriori knowledge of the intensity functions. As less knowledge is available, the processing of the 64 intensity measurements via a Kalman filter is less accurate, with less relative benefit over performance attainable with a correlator. Moreover, as discussed in the previous section, this alternate tracker embodies much more than a simple correlation tracker.

VII. Optical Processing Alternatives

Introduction

The computational burden of either of the tracking algorithms developed in the previous chapters is heavy. For that reason, a serious attempt needs to be made to develop hybrid, optical and digital, systems for implementation of those algorithms. Optical processing offers the potential of parallel processing of two-dimensional data. The two-dimensional Fourier transforms, correlations, and matrix operations which are contained in the tracking algorithms could potentially be accomplished at extremely high data rates using optical techniques. A digital computer could be programmed to perform necessary data analysis or control the optical computations. Such a hybrid system would combine the speed and parallel processing advantages of optical processors with the flexibility of digital computers.

This chapter begins with a section which describes the basic principles of optical processing and introduces the concept of hybrid processing. The next section adapts the hybrid processor to the tracking algorithms developed previously. Potential modifications to the tracking algorithms are proposed to make optical implementation easier. The purpose of this section is to present design possibilities and to provide an extremely rough, preliminary analysis of the potential for fruition of optical implementations for

each proposal.

Background

This section presents the basic principles of optical processing systems. These systems utilize optical interference to process two-dimensional incoming signals. The most common system used for optical processing is shown in Figure 27. Light from a laser point source S is collimated by lens L_C . The collimated beam of quasi-monochromatic coherent laser light is used to illuminate the object in plane P_1 and a diffraction pattern is formed. The object is placed one focal length in front of the transforming lens L_1 . The classical Fraunhofer diffraction pattern of the object's space-varying amplitude transmittance, $t_0(x_1, y_1)$, in plane P_1 is formed in the rear focal plane of lens L_1 . The complex field of the Fraunhofer diffraction pattern in plane P_2 is mathematically equal to the spatial Fourier transform (7:166) of the object.

$$T_0(x_2, y_2) = k_1 \iint t_0(x_1, y_1) \exp\left[-j\frac{2\pi}{\lambda f_1}(x_2 x_1 + y_2 y_1)\right] dx_1 dy_1 \quad (7-1)$$

$$T_0(f_x, f_y) = k_1 \iint t_0(x_1, y_1) \exp\left[-j2\pi(f_x x_1 + f_y y_1)\right] dx_1 dy_1 \quad (7-2)$$

where

k_1 = complex constant

T_0 = Fourier transform of $t_0(x_1, y_1)$

λ = mean wavelength of the laser

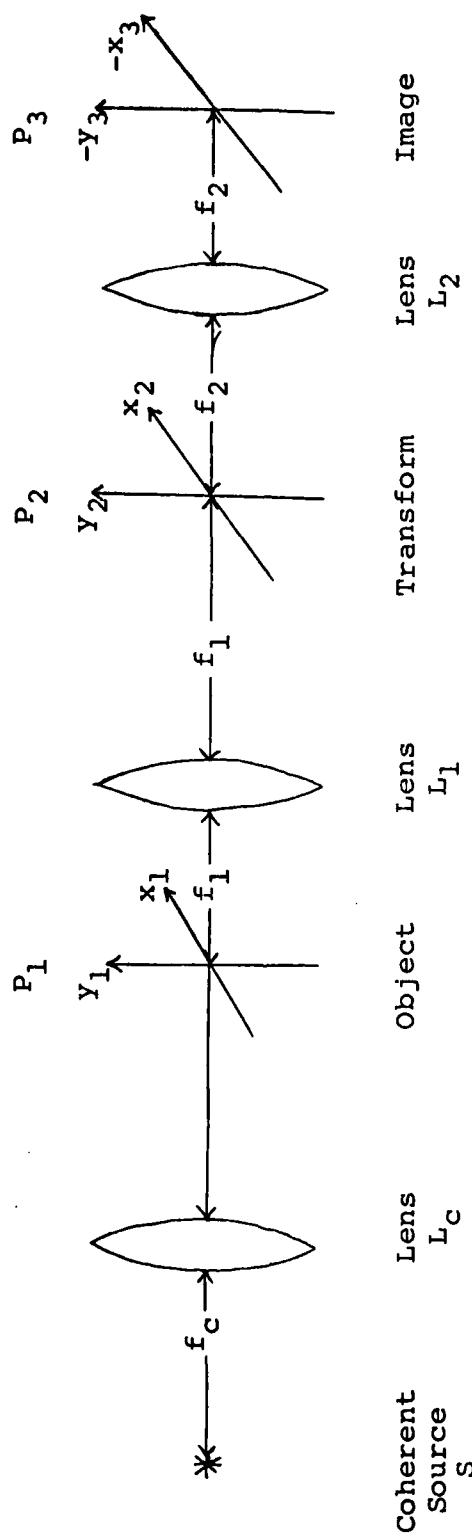


Figure 27. Basic Optical Processing System

The two-dimensional Fourier transform of the object in plane P_1 can be modified by placing an appropriate optical filter in plane P_2 . This optical filter is in general a complex function. The optical filter diffracts the light incident upon plane P_1 , producing a second diffraction pattern (14:28). This diffraction pattern is then focussed by lens L_2 onto plane P_3 .

In a hybrid system, the input plane P_1 , the transform plane P_2 , and the output image plane P_3 are all interfaced via spatial light modulators to a digital computer. To utilize the processing capability of the optical processor, some means of inputting information into and reading information out of these spatial light modulators at data rates commensurate with the 30Hz FLIR frame repetition rate is required. Spatial light modulators with cycle frequencies of 30Hz are now available from several vendors.

Itek Corporation, for example, produces a Pockels Read-out Optical Modulator, (PROM). The PROM is a spatial light modulator which is constructed from bismuth silicon oxide (bisox). Bisox is a crystal which displays both wavelength selective photoconductivity and the linear electro-optic (Pockels) effect (15:353). The complexity of the total exposure-readout process has precluded the development of a complete and exact theoretical model of PROM operation for predicting system response for varying operating conditions

(15:354). However, simplified models which are limited to particular phases of the process have been developed. These theoretical models include predictions on the wavelength dependence of PROM sensitivity, the relationship between PROM sensitivity and Bisox crystal thickness, and how PROM resolution can be increased by increasing capacitance of the blocking layer (15:364). The characteristics of the PROM, or of any of the spatial light modulators considered in this analysis do not lend themselves to presentation in a simple tabular form because of the many variables involved in the complex exposure-readout process. However, certain benchmarks on commercially available PROMs are available. A 25mm diameter is standard with special order devices with 37mm diameters also available. The desired 30Hz cycle rate is standard with a dc contrast ratio of 10^3 to $10^4:1$. A spatial frequency response of 100 cycles per millimeter has been demonstrated (15:364).

Hughes also produces a spatial light modulator which could be used in this application. The hybrid field-effect liquid crystal light valve (LCLV) is a high resolution optical-to-optical image converter. The input and output light beams are completely separated and noninteracting. The parameters which characterize LCLV performance are: sensitometry; modulation transfer function and resolution; linearity; signal-to-noise ratio; response time; optical flatness; and

image quality (16:374). The performance of the LCLV can only be given for specific values of the operational parameters (voltage bias and frequency, and orientation of the light value with respect to the output light polarization direction). The performance is particularly sensitive to the imaging light power. The use of intensifier tubes which are capable of amplifying very low light level scenes to the levels required by the light value is feasible (16:381).

These are not the only available spatial light modulators. Their characteristics are however, indicative of currently available devices.

Applying Optical Processing to Tracking

This section will apply the optical processing techniques discussed in the previous section to the two tracking algorithms developed in chapters one through six. This section will also propose potential modifications to the tracking algorithms to facilitate hybrid implementations.

The data processing algorithm of Chapter II is first considered for optical implementation. Both tracking algorithms use this procedure for generating information about the target's intensity profile. In addition, the algorithm of Figure 1 also uses the derivative information developed. The software implementation of the pattern recognition algorithm, Appendix H, along with the explanation of Chapter II clearly shows how manipulation of the amplitudes and phases

of the various spatial frequencies are used to derive the respective intensity functions. A complex filter placed in the transform plane, P_2 , of Figure 27 can be used to manipulate the amplitudes and phases of those spatial frequencies simultaneously. The spatial filtering results of the previous chapter become important here. Complex filters are usually implemented using holographic techniques which were pioneered by A. B. Vander Lugt of the University of Michigan's Radar Laboratory. The best hybrid combination for generation of the flexible complex filter which would accomplish the application of the appropriate linear phase shift to center the target's intensity function within the incoming data array requires further research. Once this filter or combination of filters is synthesized and centering of the data is accomplished, using the configuration of Figure 27 with the complex filter discussed above in plane P_2 , plane P_3 would contain the space domain representation of the centered target intensity function. The next step is to smooth this image with the previous best estimate of the intensity function stored in a spatial light modulator. Since exponentially smoothing can be accomplished in either domain, as established in Chapter VI, there is no constraint on where an optical implementation performs it. Both images are multiplied by constants (see Chapter II) and the products are added, probably via simultaneous recording techniques, to

form the new best estimate of the intensity function which is written into the spatial light modulator which stores the smoothed estimate. This smoothed estimate must also be processed by a complex shifting filter to produce the intensity function expected at the next sample time, $h(\hat{x}(t_i), t_i)$. Differentiation of this expected intensity profile can be accomplished using a simple amplitude filter in the transform plane, P_2 of Figure 27. The derivative of the object, in plane P_1 of Figure 27 can be produced by a filter function whose amplitude transmittance is

$$f(k_x) = \alpha(k_0 - k_x) \quad (7-3)$$

where

$$\alpha, k_0 = \text{constants}$$

The use of the bias term k_0 eliminates the need for a complex filter (15:361) for differentiation. This amplitude transmittance filter has its maximum transmittance at the center of the field-of-view with a minimum transmission on the edges. This is analogous to using the differentiation property of the Fourier transform as explained in Chapter II. Digitally controlled complex filters could theoretically accomplish the Chapter II intensity function derivations. Such filters are not easily implemented. The information which is to be input into the transform plane filter spatial light modulator would be computed by a digital computer. The computed desired transfer function is added to the

reference wave within the digital computer to simulate the holographic synthesis technique. The output of the digital computations is a non-negative, real-valued function which is quantized. Since the desired masks are of finite extent, recording of sampled values of the computed transfer function can completely specify its Fourier transform. This assumes that the well known sampling theorem (7:25) is satisfied. The major advantage of digitally constructed filters is that an abstract, mathematically defined filter can be computed and recorded. This eliminates the need for the availability of the point-spread function. Such complex digital computations could minimize the advantage of a hybrid implementation for a small tracking window such as the 8×8 pixels used in this research. However, if this intensity function could be shown to be only slowly changing a slower repetition rate on its generation could relieve the digital computational burden.

Another possible application of optical processing to the algorithm of Figure 1 would be in the implementation of the high dimensioned matrix operations. The parallel nature of optical processing could be used to great advantage in accomplishing these functional operations. The formation of the expected intensity function, $\underline{h}(\hat{\underline{x}}(t_i), t_i)$, was discussed above. Once the spatial domain representation of $\underline{h}(\hat{\underline{x}}(t_i), t_i)$ is created, it is to be subtracted from the measurement data $\underline{z}(t_i)$, which is the incoming FLIR data frame. The intensity

values from the FLIR could drive a spatial light modulator and then the residual could be formed by optical subtraction of $\underline{z}(t_i) - \underline{h}(\underline{x}(t_i), t_i)$ (16:383). The subtraction scheme of Reference 16, page 383, uses two LCLVs onto which $\underline{z}(t_i)$ and $\underline{h}(\hat{\underline{x}}(t_i), t_i)$ are projected. Potentially the IR radiation which was input to the FLIR could directly drive a spatial light modulator eliminating the need for a scanning type FLIR. The residual must next be multiplied by the gain matrix $K(t_i)$. In the digital implementation this matrix is 4×64 with only two unique rows. Therefore, two two-dimensional images which represent the two unique rows could be placed, via spatial light modulators, in the plane, in the space domain, which contains the residual resulting in two-dimensional images which could be sampled and summed to produce the two unique update components of Equation (4-2). These are the components which are added to our best estimate of the state before we get a measurement, $\hat{\underline{x}}(t_i^-)$, to generate $\hat{\underline{x}}(t_i^+)$.

To enhance the algorithm of Chapter V, Correlator-Kalman filter, optical correlations can be used. The transform of the template will be encoded into the filter of the transform plane, P_2 , of Figure 27. This template could theoretically be obtained by accomplishing the data processing of Chapter II optically as discussed above. A second possibility would be to derive the template digitally as was done for Chapter V and then input the information into the spatial

light modulator; this was also discussed above. A third possibility would be to array many potential templates in the filter plane and accomplish simultaneous optical correlations with all of them. Templates could be prepared in which the presence and disposition of targets were varied and where the background characteristics were also varied. The weighting of the results from the multiple template correlations could be controlled via adaptive estimation of target and environmental characteristics. The input data image would be used to modulate a collimated laser beam which is input to a transform lens. A multiple holographic lens is used as the Fourier transforming lens which directs the transform to each of the many matched filters, made from the templates. The position estimates from any of these correlation implementation options could then be processed by a Kalman filter.

Conclusion

In summary, high bandwidth optical processing in parallel with flexible lower bandwidth digital computation potentially could relieve the computational burden of the tracking algorithms developed in this research. The pattern recognition algorithm given in Chapter II would be very difficult to implement optically because of the complex filters required. For implementation of the high dimensional Kalman filter, (see Chapter IV), optical processing could accomplish the computationally intense functional matrix operations.

Optical implementation of the correlation algorithm of Chapter V has the most promise. Before any optical implementation is possible though, considerable research and development must be accomplished. The software developed in this research provides a flexible tool to determine what spatial frequency modifications enhance tracking. The zeroing out of spatial frequencies within the software is just one example of how this software can help determine what filters are needed for optical implementation of the tracking algorithms and what characteristics the resulting hybrid optical/digital algorithm would have. The testing of the differentiation filter of Equation (7-3) would just be a point by point multiplication of the spatial frequency components by an array whose values were determined by that equation. The software provided in Appendix H is well commented and easily modified for this type of testing.

VIII. Conclusions and Recommendations

Conclusions

The conclusions addressed in this section are formulated from the details of Chapter VI, Performance Analysis. The milestones which were realized in this research will be reiterated here.

As shown in Table I, the extended Kalman filter algorithm performs very well even when the intensity functions are not assumed, as was done in prior research efforts (4;5). These intensity functions were derived using the digital pattern recognition techniques developed in Chapter II. It was shown in Chapter VI that the tracking capability was not overly sensitive to increased errors in the derivation of the intensity functions. The filtering of high spatial frequencies was shown to decrease tracking errors for cases where the intensity function is relatively smooth.

In previous research efforts, apriori intensity function information was given to the extended Kalman filter. For some tracking scenarios those extended Kalman filter trackers were found to outperform correlation trackers. The alternate tracking algorithm of Chapter V was developed to combine correlation and Kalman filtering. When no apriori information on the intensity functions was assumed, the processing of the 64 intensity measurements by the Kalman filter should not outperform correlators so easily. For this reason

a correlation algorithm which makes use of dynamic model information from the Kalman filter, thresholding correlator outputs to remove false peaks prior to centroid calculations, on-line template derivation and correlator position estimate enhancement via a Kalman filter was derived. Chapter VI showed how this algorithm outperforms the originally envisioned algorithm of Figure 1.

Finally Chapter VII presented the optical processing alternatives for implementing the tracking algorithms. The parallel nature of optical processors provide the potential to relieve the computational burden of the tracking algorithms. Prior to any optical implementation, extended research and development must be accomplished.

Recommendations

Further research is recommended to assess the effects of incorporating the IR laser's FLIR image in the model's used. The use of optical windows could potentially notch the contribution of the IR laser out of the FLIR image. The laser will however induce hot spots on the target which were not part of the original multi-hotspot target model. Modern estimation techniques could also be used to determine what translational or phase distortion effects occurred during propagation of the laser beam. The use of spatial light modulators could potentially obtain the phase distortion induced by the atmosphere. A compensating imposed distortion on the

outgoing beam could then maximize target damage.

Another area of suggested research is to attempt optical implementation of those portions of the tracking algorithms discussed in Chapter VII. This must include research into the feasibility of using the many types of available spatial light modulators.

The alternative tracking algorithm developed in Chapter V needs to be investigated further. The optimal thresholding and peak detection or centroiding techniques could further improve tracking ability. Adaptive techniques for on-line tuning could also be investigated.

The algorithms of this research should now be tested using real FLIR data. A combined Physics-EE effort at AFIT would be required to assemble the experimental apparatus needed to generate the real FLIR data. As an alternative, the data could potentially be provided to AFIT by the Air Force Weapons Laboratory.

The use of different dynamics models should also be investigated. Adaptive models or models compatible with different tracking scenarios could also be investigated.

An investigation of how these algorithms perform when tracking a slowly changing target shape is also needed. The variation of alpha to enhance tracking of slowly changing targets could also be investigated.

Finally, only empirical convergence of the algorithms

developed was shown during this research. A theoretical proof of convergence of these adaptive estimation techniques should be derived.

Bibliography

1. DeMott, John S., "A Technology to Transform War," TIME: 87-9 (s 15'80).
2. Robinson, Clarence A. Jr., "Beam-Target Interaction Tested," Aviation Week and Space Technology: 114, 14-17 (0'8'79).
3. Robinson, Clarence A. Jr., "Laser Technology Demonstrated," Aviation Week and Space Technology: 16-19 (F'16'81).
4. Singletary, James Jr., "Adaptive Laser Pointing and Tracking Problem," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1980.
5. Mercier, Daniel E., "An Extended Kalman Filter for Use in a Shared Aperture Medium Range Tracker," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1978.
6. Harnly, Douglas A. and Jensen, Robert L., "An Adaptive Distributed-Measurement Extended Kalman Filter for a Short Range Tracker," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1979.
7. Goodman, Joseph W. Introduction to Fourier Optics. San Francisco: McGraw-Hill Book Company, 1968.
8. Oppenheim, Alan V. and Schaffer, Ronald W. Digital Signal Processing. New Jersey: Prentice-Hall Incorporated, 1975.
9. Oppenheim, Alan V. Applications of Digital Signal Processing. New Jersey: Prentice-Hall Incorporated, 1978.
10. Bedworth, David D. Industrial Systems Planning, Analysis, Control. New York: The Ronald Press Company, 1973.
11. The Analytic Sciences Corporation. Advanced Adaptive Optics Control Techniques. TR-996-1. Prepared for the Air Force Weapons Laboratory, Kirtland Air Force Base, New Mexico, January 6, 1978.
12. Hogge, C. B. and Butts, R. R. "Frequency Spectra for the Geometric Representation of Wavefront Distortions Due to Atmospheric Turbulence," IEEE Transactions on Antenna and Propagation, Vol. AP-24, No. 2, March 1976.

13. Maybeck, Peter S. Stochastic Models, Estimation, and Control Volume I. New York: Academic Press Incorporated, 1979.
14. Casasent, David. "Pattern Recognition: a review," IEEE Spectrum: 28-33 (March '81).
15. Horwitz, Bruce A. and Corbett, Francis J., "The PROM - Theory and Applications for the Pockels Readout Optical Modulator," Optical Engineering: 353-364 (July - August '78).
16. Bleha, Lipton, Weiner-Avnear, Grinberg, Reif, Casasent, Brown, Markevitch, "Application of the Liquid Crystal Light Valve to Real-Time Optical Data Processing," Optical Engineering: 371-384 (July - August '78).
17. Maybeck, Peter S. Unpublished lecture materials distributed in EE 7.66, Stochastic Estimation and Control II. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, 1976.

Appendix A

Two-Dimensional Finite Discrete Fourier Transform Interpretation and Test

Similar to the presentation of J. W. Goodman, reference 7, the Rect function is defined to extract one period of the spatial intensity function as

$$\text{Rect}_N(x, y) = \begin{cases} 1 & 0 \leq x \leq N-1, 0 \leq y \leq N-1 \\ 0 & \text{otherwise} \end{cases} \quad (\text{A-1})$$

Reproducing Equations (2-3) and (2-4) of Chapter II and using the Rect function to define the area sequence $g(x, y)$ as being zero outside the interval $0 \leq x \leq N-1$

$$g(x, y) = g'(x, y) \text{Rect}_N(x, y) \quad (\text{A-2})$$

$$G(f_x, f_y) = \left[\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} g(x, y) e^{-j\frac{2\pi}{N}(f_x x + f_y y)} \right] \text{Rect}_N(f_x, f_y) \quad (\text{A-3})$$

$$g(x, y) = \frac{1}{N^2} \left[\sum_{f_x=0}^{N-1} \sum_{f_y=0}^{N-1} G(f_x, f_y) e^{j\frac{2\pi}{N}(f_x x + f_y y)} \right] \text{Rect}_N(f_x, f_y) \quad (\text{A-4})$$

$G(f_x, f_y)$ is the Finite Discrete Fourier Transform of $g(x, y)$ (8:118). The Rect function of Equation (A-1) is separable in the two independent variables:

$$\text{Rect}_N(f_x, f_y) = \text{Rect}_N(f_x) \text{Rect}_N(f_y) \quad (\text{A-5})$$

Equation (A-3) can now be written as

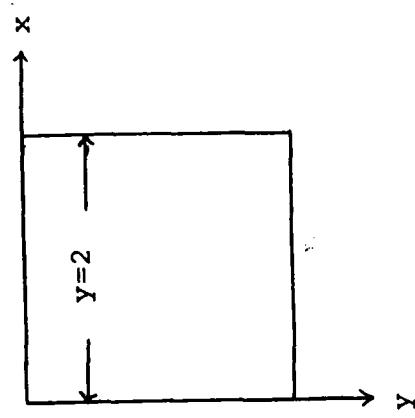
$$G(f_x, f_y) = \left[\sum_{y=0}^{N-1} X(f_x, y) \exp \frac{-j2\pi}{N} (f_y y) \right] \text{Rect}_N(f_y) \quad (A-6)$$

where

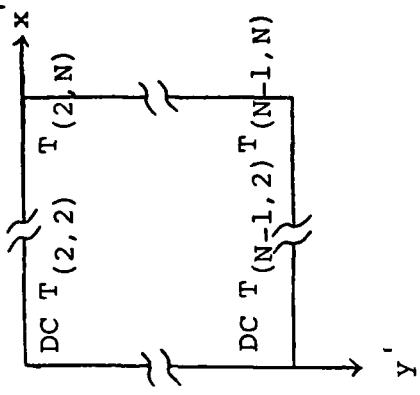
$$X(f_x, y) = \left[\sum_{x=0}^{N-1} g(x, y) \exp \frac{-j2\pi}{N} (f_x x) \right] \text{Rect}_N(f_x) \quad (A-7)$$

Equation (A-7), $X(f_x, y)$ corresponds to an N-point one dimensional Discrete Fourier Transform for each value of the row index y (8:118). $X(f_x, y)$ is the result of N one dimensional transforms, one for each row of $g(x, y)$. In Figure A-1a, if y is held constant, say $y=2$, and a one-dimensional Fourier Transform is accomplished across that row, the variation in the image intensities from column to column would result in the zero frequency of that variation to be located at $x'=1$ and $y'=2$, of Figure A-1b, while the coefficient associated with the fundamental, the first harmonic, in both directions would be found at $x'=2$ and $y'=2$ with its conjugate located at $x'=N$ and $y'=2$.

To complete the two-dimensional Discrete Fourier Transform Equation (A-6) expresses how to implement the N one-dimensional transforms along each column. To summarize this general interpretation of the two-dimensional Discrete Fourier Transform, the Equations (A-6) and (A-7) show how the two-dimensional transform is achieved by using a one-dimensional transform on the rows first and then on the columns, or vice versa.



(a) Original Data Array



(b) Transformed Data

Figure A-1. Row Transform Result of 2-D DFT

For the case where the image intensity function is separable in the two independent variables x and y , for example, $g(x, y)$ having the property that

$$g(x, y) = g_1(x)g_2(y) \quad (A-8)$$

the two-dimensional Discrete Fourier Transform, $G(f_x, f_y)$, becomes the product of the one-dimensional independent transforms $G_1(f_x)$ and $G_2(f_y)$.

$$G(f_x, f_y) = G_1(f_x)G_2(f_y) \quad (A-9)$$

A function of two independent variables is separable within a specific coordinate system if it can be written as a product of two functions of one independent variable each. The two-dimensional transform degenerates to holding the row index constant, for example, and running the one-dimensional transform across the columns, for anyone of the N rows. Similarly the column index is held while the one-dimensional transform is accomplished across the rows, and this is done for any of the columns. The results of these two independent transforms are then multiplied together.

This leads to some interesting algebra which can be exploited to test the implementation of the two-dimensional Discrete Fourier Transform. In Figure A-1, $T_{(2,2)}$ will consist of the product of the two one-dimensional transform fundamental coefficients for that row or column. If x is the column coordinate and y is the row coordinate, then

$T(y, x)$ is defined as

$$T(2,2) = \left[\begin{bmatrix} 1^{\text{st}} \text{ harmonic in } y \\ \text{for column 2} \end{bmatrix} \triangleq y_{(1,2)} \right] \cdot \left[\begin{bmatrix} 1^{\text{st}} \text{ harmonic in } x \\ \text{for row 2} \end{bmatrix} \triangleq x_{(1,2)} \right] \quad (\text{A-10})$$

$$T(2,N) = \left[\begin{bmatrix} 1^{\text{st}} \text{ harmonic in } y \\ \text{for column N} \end{bmatrix} \triangleq y_{(1,N)} \right] \cdot \left[\begin{bmatrix} \text{conjugate of } 1^{\text{st}} \\ \text{harmonic in } x \\ \text{for row 2} \end{bmatrix} \triangleq x_{(1,2)}^* \right] \quad (\text{A-11})$$

$$T(N-1,2) = \left[\begin{bmatrix} \text{conjugate of } 1^{\text{st}} \\ \text{harmonic in } y \\ \text{for column 2} \end{bmatrix} \triangleq y_{(1,2)}^* \right] \cdot \left[\begin{bmatrix} 1^{\text{st}} \text{ harmonic in } x \\ \text{for row } N-1 \end{bmatrix} \triangleq x_{(1,N-1)} \right] \quad (\text{A-12})$$

$$T(N-1,N) = \left[\begin{bmatrix} \text{conjugate of } 1^{\text{st}} \\ \text{harmonic in } y \\ \text{for column N} \end{bmatrix} \triangleq y_{(1,N)}^* \right] \cdot \left[\begin{bmatrix} \text{conjugate of } 1^{\text{st}} \\ \text{harmonic in } x \\ \text{for row } N-1 \end{bmatrix} \triangleq x_{(1,N-1)}^* \right] \quad (\text{A-13})$$

if $y_1 < N/2, x_1 < N/2$

$$T(y_1, x_1) = \left[\begin{bmatrix} (y_1-1) \text{ harmonic in } y \\ \text{for column } x_1 \end{bmatrix} \triangleq y_{(y_1-1, x_1)} \right] \cdot \left[\begin{bmatrix} (x_1-1) \text{ harmonic in } x \\ \text{for row } y_1 \end{bmatrix} \triangleq x_{(x_1-1, y_1)} \right]$$

If the variation in intensity across every row is equal to the variation in intensity for every other row and the variation across all of the columns is similarly set equal, then the components of the transform can be defined as

$$\begin{aligned}
 y_{(1,2)} &= a + jb & x_{(1,2)} &= c + jd \\
 y_{(1,N)} &= a + jb & x_{(1,N-1)} &= c + jd \\
 y_{(1,2)}^* &= a - jb & x_{(1,2)}^* &= c - jd \\
 y_{(1,N)}^* &= a - jb & x_{(1,N-1)} &= c - jd
 \end{aligned} \tag{A-14}$$

Using Equation (A-14) to solve (A-10) through (A-13)

$$T_{(2,2)} = y_{(1,2)} \cdot x_{(1,2)} = (a+jb)(c+jd) = (ac-bd) + j(bc+ad) \tag{A-15}$$

$$T_{(2,N)} = y_{(1,N)} \cdot x^*_{(1,2)} = (a+jb)(c-jd) = (ac+bd) - j(ad-bc) \tag{A-16}$$

$$T_{(N-1,2)} = y^*_{(1,2)} \cdot x_{(1,N-1)} = (a-jb) \cdot (c+jd) = (ac+bd) + j(ad-bc) \tag{A-17}$$

$$T_{(N-1,N)} = y^*_{(1,N)} \cdot x^*_{(1,N-1)} = (a-jb) \cdot (c-jd) = (ac-bd) - j(ad+bc) \tag{A-18}$$

Equation (A-15) for $T_{(2,2)}$ is the conjugate of Equation (A-18) for $T_{(N-1,N)}$ under the conditions of uniform variations on any row and similarly for the columns, which just implies separability.

To implement the two-dimensional Fourier Transform, subroutine Fourt was used, which is a common Fortran subroutine (4:). To test subroutine Fourt to see where it places harmonics with a two-dimensional array, the results of Equations (A-15) through (A-18) were used. Figures A-2 through A-7 show the results of the tests.

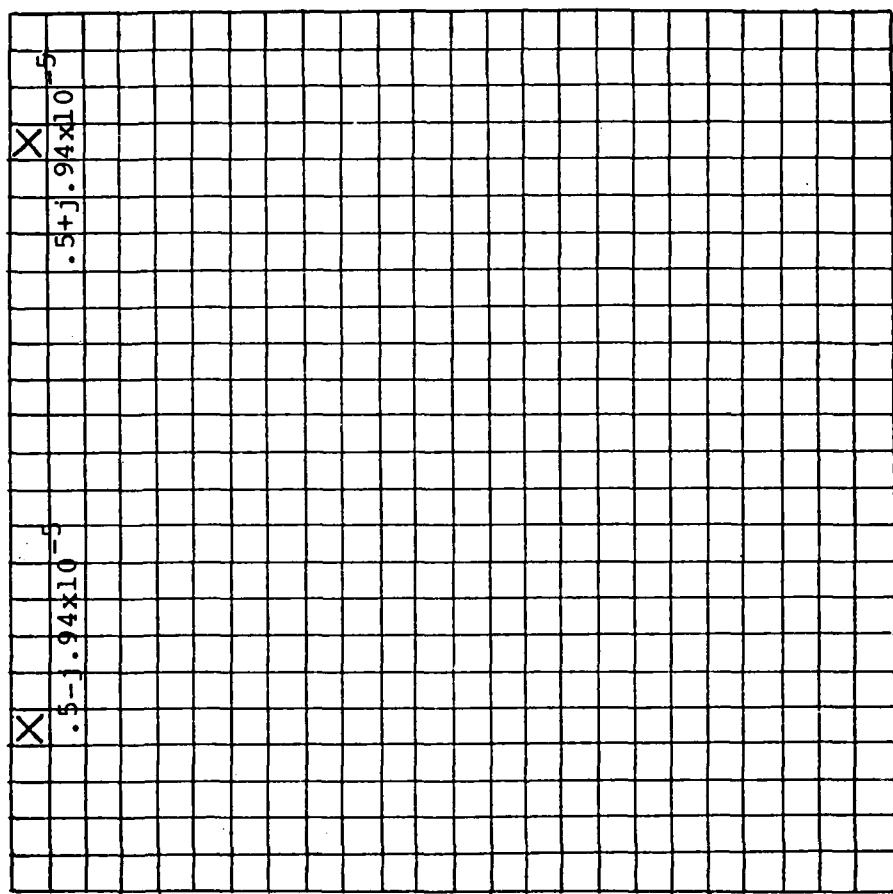


Figure A-2. $\cos 2\pi x/6$

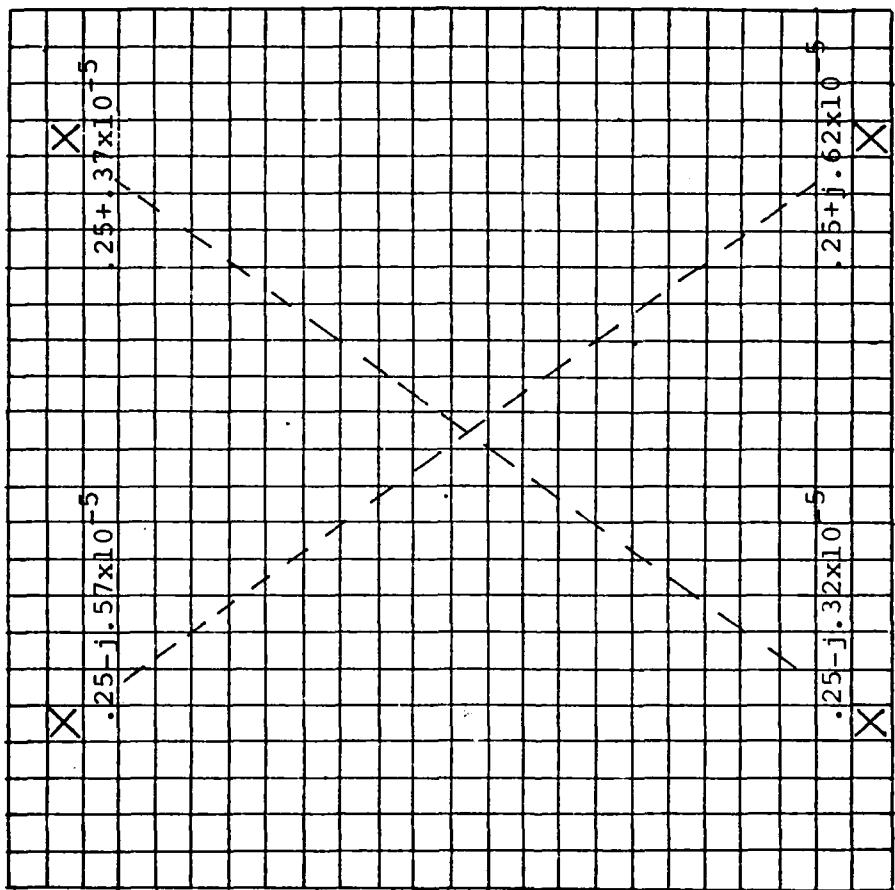


Figure A-3. $(\cos 2\pi x/6) (\cos 2\pi y/24)$

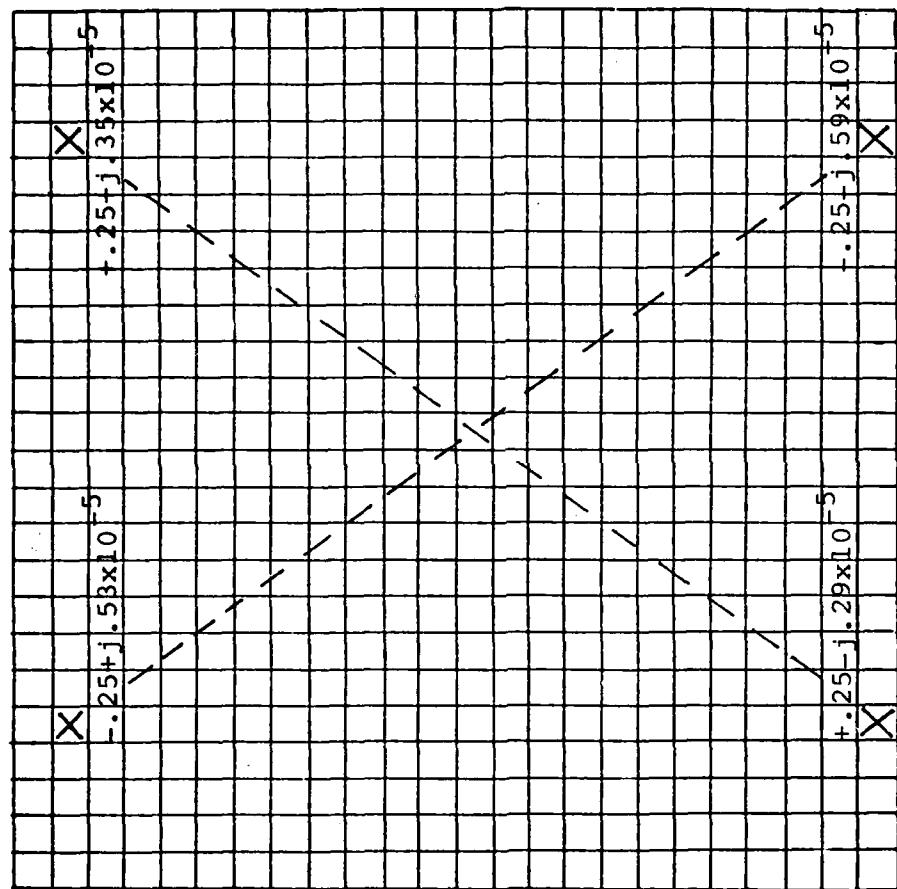


Figure A-4. $(\sin 2\pi x/6)$ ($\sin 2\pi y/24$)

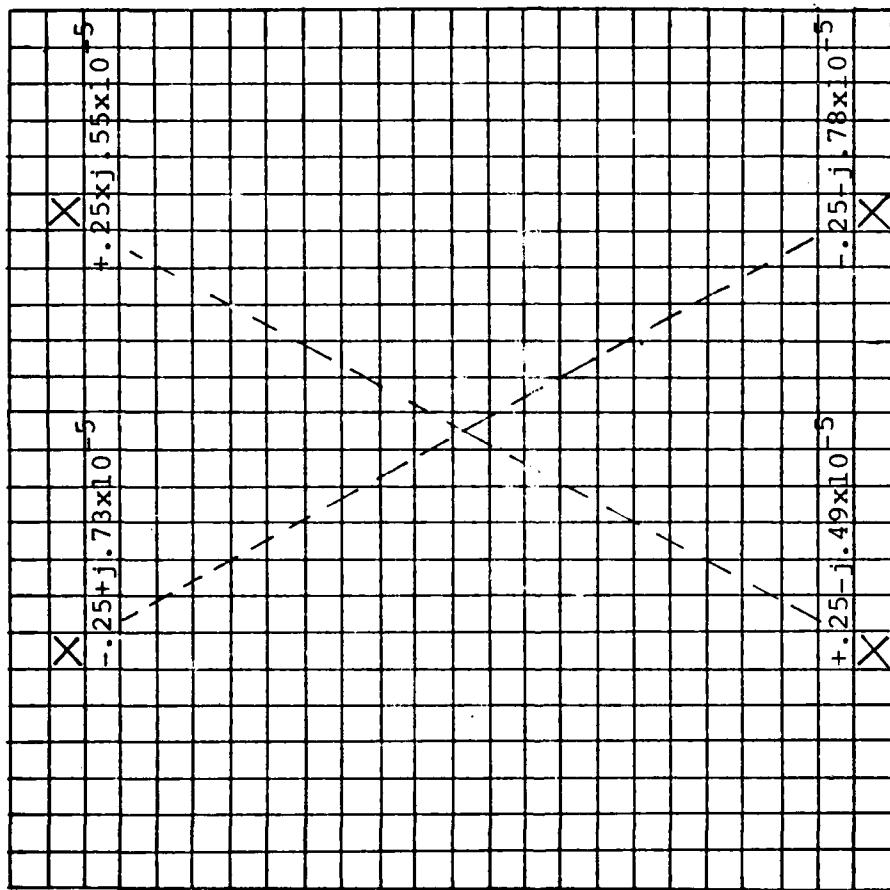


Figure A-5. $(\sin 2\pi x/4) (\sin 2\pi y/24)$

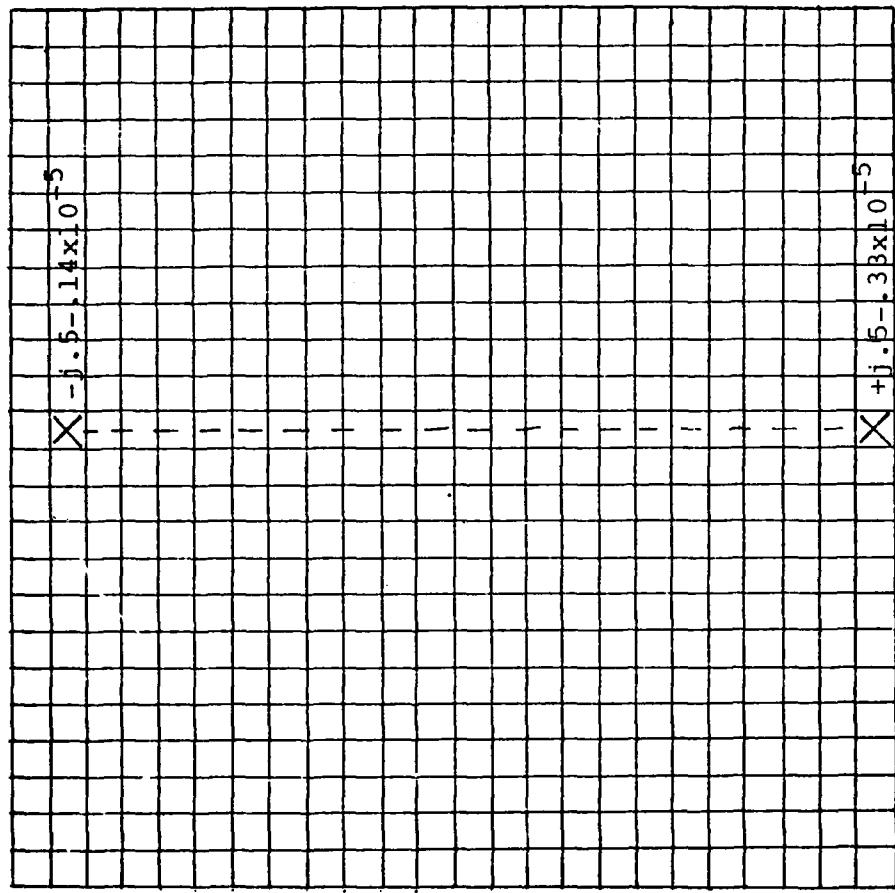


Figure A-6. $(\cos 2\pi x/2) (\sin 2\pi y/24)$

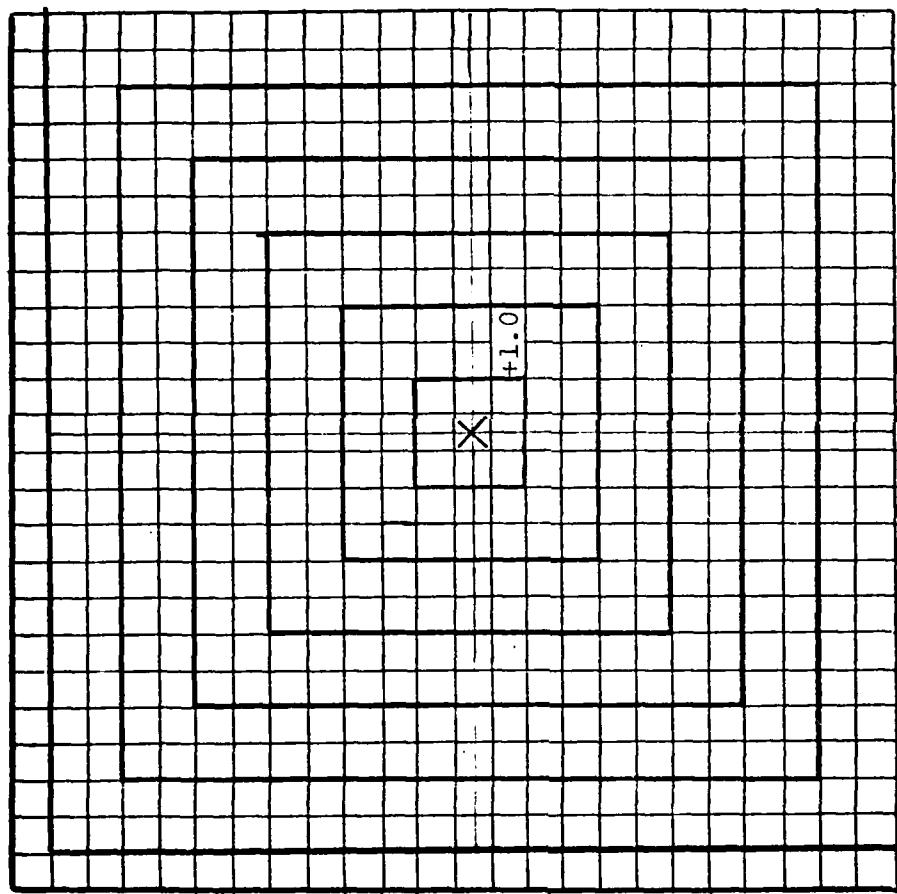


Figure A-7. $(\cos 2\pi x/2)(\cos 2\pi y/2)$

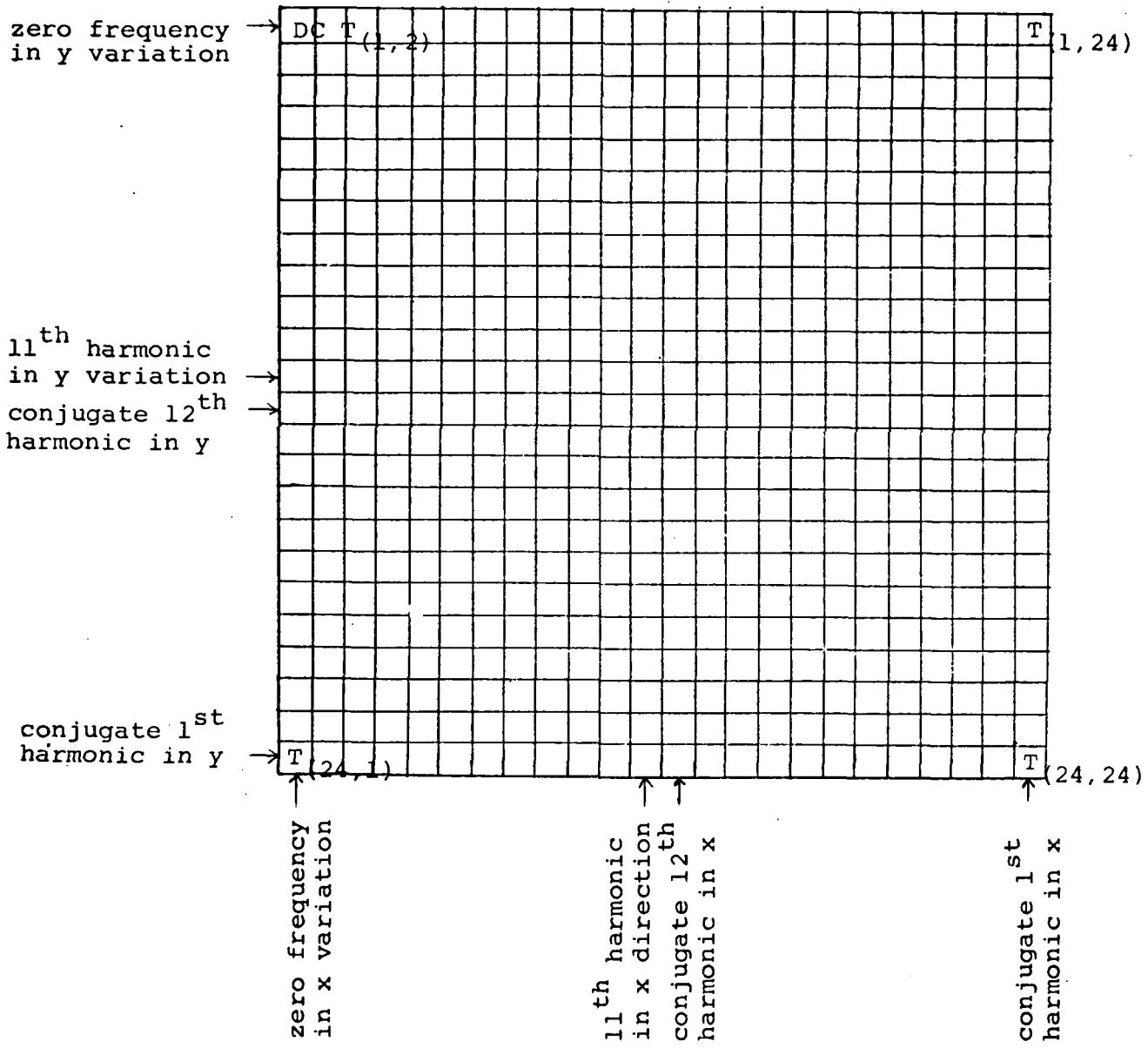
Figure A-1 had as its input a 24 x 24 array with magnitudes of each element varying only across the rows via $\cos(2\pi x/6)$. Using the above interpretation of a two-dimensional Fourier Transform, the fundamental frequency assumed would be one which corresponded to one period across the 24 element array. Since the input obviously fits 4 cycles in that space, the only nonzero component of the Fourier Transform expected would be at the fourth harmonic. The transform would also be expected to be real only since the input is a pure cosine. Figure A-2 is then encouraging in that the only nonzero components are where expected and the imaginary portions of that answer is extremely small. The reason that there are nonzero elements only in the first row is that when the transform was accomplished across the columns, there was no variation in the y coordinate, which is equivalent to taking the transform of a constant one. That transform results in a one in the zero frequency components in each column which is the first row and zeros everyplace else. When the multiplication of Equation (A-9) is accomplished, Figure A-2 results.

Figure A-3 had an input of $(\cos 2\pi x/6) \cdot (\cos 2\pi y/24)$ which is the same variation in the x-direction and a variation in the y-direction which corresponds to one period exactly. The only nonzero result from the transform along the columns, variation in y, should be in the fundamental, which it is.

Figure A-4 through A-7 are just further examples to demonstrate Fourt. Figure A-4 had an input of the fourth harmonic in the x-direction and the fundamental in the y-direction. Figure A-5 contains the sixth harmonic in the x-direction. Figure A-6 contains the twelfth harmonic in the x-direction which only appears as a conjugate. Figure A-7 contains only the twelfth harmonic in both directions and encloses spatial frequencies within boxes. A thorough understanding of these results is needed to understand taking derivatives and shifting in the transform domain.

In summary Figure A-8 is provided to show the output of the subroutine Fourt. The DC component is the product of the zero frequency components of the one-dimensional transforms in both directions. Elements $T_{(1,2)}$ through $T_{(24,24)}$ can be interpreted similarly to Equations (A-10) through (A-13).

Figure A-8. Output of Fourt



Appendix B

Shifting Property of the Two-Dimensional Finite Discrete Fourier Transform

To accomplish interframe filtering, the target's intensity profile must be centered within each data frame. This would allow the successively centered frames to be averaged to attenuate noise. However, for the reasons presented in Chapter I, the true target's intensity profile is offset from the center of the field-of-view. This offset from the center of the data array is estimated by the Kalman filter updated states, $\hat{x}(t_i^+)$. The problem then becomes how to manipulate the data array so that the target's intensity profile is centered within the field-of-view. The solution is to use the shifting property of the Fourier Transform to alter the data to negate a spatial shift of an amount calculated from the filter's updated estimate.

As stated in Chapter II, a shift of a function in the space domain introduces a linear phase shift in the frequency domain (8:9). Equation (8) of Chapter II is now rewritten to include the summations of the Two-Dimensional Finite Discrete Fourier Transform.

$$F \left[g(x-x_0, y-y_0) \right] = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} g(x, y) \exp \left[-j \frac{2\pi}{N} (f_x x + f_y y) \right] \exp \left[-j \frac{2\pi}{N} (f_x x_0 + f_y y_0) \right]$$
$$\quad \quad \quad 0 \leq y_0 \leq N \quad \quad \quad 0 \leq x_0 \leq N$$

(B-1)

This equation can be shortened, as done with Equation (8) of Chapter II by substituting $G(f_x, f_y)$ for the Fourier Transform of the unshifted function $g(x, y)$.

$$F\left[g(x-x_0, y-y_0)\right] = G(f_x, f_y) \exp\left[\frac{-j2\pi}{N}(f_x x_0 + f_y y_0)\right] \quad 0 \leq y_0 \leq N \quad 0 \leq x_0 \leq N \quad (B-2)$$

The reason for the qualifier of shift magnitudes under Equations (B-1) and (B-2) is that if a shift were allowed to be greater than the period assumed by the Fourier Transform, see Appendix A, it could not be distinguished from a shorter shift. This is because of aliasing. Therefore, $x_0 \leq N$ and $y_0 \leq N$ are required for the interpretation of the shift as a rotation in two dimensions to be a valid unique representation.

The implication of Equations (B-1) and (B-2) using different factors for different spatial frequencies is that the results of Appendix A, Two-Dimensional Fourier Transforms, must be used to locate each of the spatial frequencies within the transform domain. To implement the shift, each point in the transform domain must be multiplied by the conjugate of the phase shift induced by the spatial translation.

$$\begin{aligned} g(x, y) &= F^{-1}\left[G(f_x, f_y) \exp\left[\frac{-j2\pi}{N}(f_x x_0 + f_y y_0)\right] \exp\left[\frac{+j2\pi}{N}(f_x x_0 + f_y y_0)\right]\right] \\ &= F^{-1}\left[F\left[g(x-x_0, y-y_0)\right] \exp\left[\frac{+j2\pi}{N}(f_x x_0 + f_y y_0)\right]\right] \quad (B-3) \end{aligned}$$

The centered intensity function $g(x, y)$ can be derived from the inverse Fourier Transform of the Fourier Transform of the shifted function $F[g(x-x_0, y-y_0)]$ via Equation (B-3). The shifted intensity function $g(x-x_0, y-y_0)$ is the input frame of data in Figure 1 of Chapter I. The Two-Dimensional Finite Discrete Fourier Transform of this data is then taken, which is $F[g(x-x_0, y-y_0)]$. In Appendix A, Figure A-8, the output of a two-dimensional transform is shown. To implement a shift each point of the two-dimensional transformed data array must be multiplied by the conjugate of the translation induced linear phase shift as shown in (B-3).

Subroutine Shift shifts a data array by an amount specified by its parameters. In Figure A-8, for example, point $T_{(1,2)}$ is the point of the transformed array in the first row second column. Figure A-8 shows it is the first harmonic in the x -variation and the zero frequency in the y -variation (i.e., $f_x=1, f_y=0$). This point would therefore be multiplied by $\exp[-\frac{i2\pi}{N}x_0]$ to implement a shift of x_0, y_0 . Taking another point, for example, $T_{(24,24)}$ is the transformed data array component in the twenty-fourth row twenty-fourth column. Figure A-8 shows this point to be the product of the conjugate of the first harmonics in both directions. To implement a shift this point would be multiplied by $\exp[\frac{+i2\pi}{N}(x_0+y_0)]$.

Figure B-1a and B-1b show the results of subroutine Shift. The inputs to Shift are the data array, the dimension

1	1
1	1

(a) Original data

1	1
1	1

(b) x shift = 2., y shift = 2.

Figure B-1. Results of Subroutine Shift

of that data array, and the amount in pixels of shift desired. This routine was tested with data representing a single Gaussian and a multi-hotspot three Gaussian profile.

In summary, Equation (B-3) shows how the centered intensity function can be derived from the Fourier Transform of the spatially translated intensity function $F[g(x-x_0, y-y_0)]$. The Two-Dimensional Finite Discrete Fourier Transform assumes that this finite-area array represents one period of a two-dimensional periodic sequence. For this reason, if a shift is greater than the period it can't be distinguished from a shorter shift.

Appendix C

Implementation of the Exponential Smoothing Algorithm

As explained in Chapter II, the target's intensity pattern is corrupted by noise. Interframe smoothing of centered intensity patterns is accomplished with an exponential smoothing algorithm defined by

$$\hat{y}(t) = \alpha \underline{y}(t) + (\alpha-1) \hat{y}(t-1) \quad (C-1)$$

where

$\hat{y}(t)$ = current averaged data frame

$\underline{y}(t)$ = current data frame

$\hat{y}(t-1)$ = previous result of averaging data

α = smoothing constant $0 \leq \alpha \leq 1$

For smoothing in the frequency domain Equation (C-1) becomes

$$\hat{L}(t_i) = \alpha \underline{L}(t_i) + (1-\alpha) \hat{L}(t_{i-1}) \quad (C-2)$$

where

$\hat{L}(t_i)$ = current averaged data frame in frequency domain

$\underline{L}(t_i)$ = transform of current noise corrupted data frame

$\hat{L}(t_{i-1})$ = previous result of averaging in frequency domain

α = smoothing constant $0 \leq \alpha \leq 1$

If, for example, the steady state value of α is to be 0.1, this algorithm would be implemented as follows. The smoothing constant, α , is varied for the first ten frames

$(\alpha=1/k \ k=1, 2, 3, \dots, 10)$ until the steady state value of $\alpha = .1$ is reached. Once steady state is reached Equation (C-2) becomes:

$$\hat{L}(t_i) = .1 \underline{L}(t_i) + .9 \hat{L}(t_{i-1}) \quad (C-3)$$

when the smoothing constant, α , is set to $1/k$ for the first ten frames, k is equal to from one to ten, the equations below describe the smoothing algorithm.

$$\begin{aligned} k = 1 \quad \hat{L}(t_1) &= \underline{L}(t_1) \\ k = 2 \quad \hat{L}(t_2) &= \frac{1}{2}\underline{L}(t_2) + \frac{1}{2}\hat{L}(t_1) = \frac{1}{2}\underline{L}(t_2) + \frac{1}{2}\underline{L}(t_1) \\ k = 3 \quad \hat{L}(t_3) &= \frac{1}{3}\underline{L}(t_3) + \frac{2}{3}\hat{L}(t_2) = \frac{1}{3}\underline{L}(t_3) + \frac{2}{3}[\frac{1}{2}\underline{L}(t_2) + \frac{1}{2}\underline{L}(t_1)] \\ &= \frac{1}{3}\underline{L}(t_3) + \frac{1}{3}[\underline{L}(t_2) + \underline{L}(t_1)] \\ k = 4 \quad \hat{L}(t_4) &= \frac{1}{4}\underline{L}(t_4) + \frac{3}{4}\hat{L}(t_3) \\ &= \frac{1}{4}\underline{L}(t_4) + \frac{3}{4}[\frac{1}{3}(\underline{L}(t_3) + \underline{L}(t_2) + \underline{L}(t_1))] \\ &= \frac{1}{4}[\underline{L}(t_4) + \underline{L}(t_3) + \underline{L}(t_2) + \underline{L}(t_1)] \quad (C-4) \end{aligned}$$

This time averaging of the first k frames continues until the steady state value of $\alpha = .1$ and Equation (C-3) is reached. For the tenth frame the smoothing algorithm becomes:

$$\begin{aligned} k = 10 \quad \hat{L}(t_{10}) &= \frac{1}{10}\underline{L}(t_{10}) + \frac{9}{10}[\frac{1}{9}(\underline{L}(t_9) + \dots + \underline{L}(t_1))] \\ &= \frac{1}{10}[\underline{L}(t_{10}) + \dots + \underline{L}(t_1)] \end{aligned}$$

$$\begin{aligned}
 k = 11 \quad \hat{L}(t_{11}) &= \frac{1}{10} L(t_{11}) + (.9)(.1) \cdot \sum_{i=1}^{10} L(t_i) \\
 k = 12 \quad \hat{L}(t_{12}) &= \frac{1}{10} L(t_{12}) + (.9)(.1) L(t_{11}) + (.9)(.9)(.1) \cdot \\
 &\quad \sum_{i=1}^{10} L(t_i)
 \end{aligned} \tag{C-5}$$

For $k \geq 11$ the exponential smoothing equation can be generalized to the following equation

$$\begin{aligned}
 k \geq 12 \quad \hat{L}(t_k) &= \frac{1}{10} L(t_k) + \sum_{l=1}^{k-11} (.9)^l (.1) L_{k-l} \\
 &\quad + (.9)^{k-10} (.1) [L(t_{10}) + \dots + L(t_1)]
 \end{aligned} \tag{C-6}$$

When $k=11$ the second term of Equation (C-6), the summation from $l=1$ to $k-11$, becomes zero since the upper limit on the summation is less than the lower limit.

If it is desired to do this algorithm in the spatial domain the same summations hold.

$$\begin{aligned}
 i \geq 11 \quad \hat{h}(x, y, t_i) &= .1 h(x, y, t_i) + \sum_{k=1}^{i-11} (.9)^k (.1) h(x, y, t_{i-k}) \\
 &\quad + (.9)^{i-10} (.1) [h(x, y, t_{10}) + \dots + h(x, y, t_1)] \\
 i \leq 10 \quad h(x, y, t_i) &= F^{-1}[L(t_i)] = \frac{1}{i} \sum_{k=1}^i h(x, y, t_k)
 \end{aligned} \tag{C-7}$$

In summary, an exponential smoothing algorithm was used to minimize the effects of the corrupting noise. Equations (C-4), (C-5), and (C-6) express the implementation of

this algorithm in the frequency domain. This same algorithm
is valid in the spatial domain as shown in Equation (C-7).

Appendix D

Implementation of the Spatial Derivatives Using Fourier Transforms

The extended Kalman filter algorithm requires the spatial derivatives of the two-dimensional intensity function with respect to each of the states (see Chapter IV). The derivative property of the Fourier Transform can be used to obtain these derivatives.

$$\begin{aligned} F\left[\frac{\partial h(x,y)}{\partial x}\right] &= j2\pi f_x \cdot F[h(x,y)] \\ F\left[\frac{\partial h(x,y)}{\partial y}\right] &= j2\pi f_y \cdot F[h(x,y)] \end{aligned} \quad (D-1)$$

Equation (D-1) shows that differentiation in the x direction of the space domain is equivalent to multiplication by $j2\pi(f_x)$ in the frequency domain and similarly for the y direction. The derivative of the intensity function can be expressed in terms of the transform of that function.

To implement Equation (D-1), the results of Appendix A must be used to locate each of the spatial frequencies in the transform domain. To obtain the spatial derivative of the intensity function, each data point in the transform domain must be multiplied by $j2\pi$ (corresponding spatial frequency in the direction of the desired derivative). In Figure A-8, for example point $T_{(1,2)}$ is the point of the transformed array in the first row and second column. Figure A-8 shows this point to be the product of the first harmonic

in the x-variation and the zero frequency in the y-variation.

To obtain the spatial derivative in the x-spatial-direction, this point would be multiplied by $(j2\pi/N)$. Taking another point, say $T_{(24,24)}$, which is the transformed data array component in the twenty-fourth row and twenty-fourth column, the conjugate of spatial frequencies can be demonstrated.

$T_{(24,24)}$ is the product of the conjugate of the first harmonics in both x and y directions. To obtain the spatial derivative in the x-direction this point would be multiplied by $-j2\pi/N$.

To show how the Derivative algorithm was tested Figures D-1 through D-3 are included. Figure D-1 shows the original data to be sinusoids in both spatial directions, of one period in x and two periods in y, $\sin(2\pi x/24) \cdot \sin(2\pi y/12)$. Figure D-2 shows the result of using the Derivative Property of the Fourier Transform to obtain the spatial derivative in the x-direction. The result is a cosinusoid of one period. Similarly, Figure D-3 shows the spatial derivative in the y-direction. The result is a cosinusoid of two periods. These figures use a grey-scale routine which attempts to show increases in intensity by more over-prints. (see Subroutine Display). This routine only reflects the absolute values of intensities.

In summary, Equation D-1 shows how the spatial derivative can be expressed in terms of the transform of that in-

```
-----  
! +X00000X+ +X00000X+ !  
! +000000000+ +000000000+!  
! +000000000+ +000000000+!  
! +000000000+ +000000000+!  
! +X00000X+ +X00000X+ !  
!  
! +X00000X+ +X00000X+ !  
! +000000000+ +000000000+!  
! +000000000+ +000000000+!  
! +000000000+ +000000000+!  
! +X00000X+ +X00000X+ !  
!  
! +X00000X+ +X00000X+ !  
! +000000000+ +000000000+!  
! +000000000+ +000000000+!  
! +000000000+ +000000000+!  
! +X00000X+ +X00000X+ !  
!
```

Figure D-1. Original Data $\sin \frac{2\pi}{24} x \cdot \sin \frac{2\pi}{24} y$

```
!
!000X+ +X00000X+ +X00!
!00000+ +000000000+ +0000!
!00000+ +000000000+ +0000!
!00000+ +000000000+ +0000!
!000X+ +X00000X+ +X00!
!
!000X+ +X00000X+ +X00!
!00000+ +000000000+ +0000!
!00000+ +000000000+ +0000!
!00000+ +000000000+ +0000!
!000X+ +X00000X+ +X00!
!
!000X+ +X00000X+ +X00!
!00000+ +000000000+ +0000!
!00000+ +000000000+ +0000!
!00000+ +000000000+ +0000!
!000X+ +X00000X+ +X00!
!
!000X+ +X00000X+ +X00!
!00000+ +000000000+ +0000!
!00000+ +000000000+ +0000!
!00000+ +000000000+ +0000!
!000X+ +X00000X+ +X00!
```

Figure D-2. Spatial Derivative in x

```
-----  
! +00000000+ +00000000+!  
! +00000000+ +00000000+!  
! +X00000X+ +X00000X+ !  
!  
! +X00000X+ +X00000X+ !  
! +00000000+ +00000000+!  
! +00000000+ +00000000+!  
! +00000000+ +00000000+!  
! +X00000X+ +X00000X+ !  
!  
! +X00000X+ +X00000X+ !  
! +00000000+ +00000000+!  
! +00000000+ +00000000+!  
! +00000000+ +00000000+!  
! +X00000X+ +X00000X+ !  
!  
! +X00000X+ +X00000X+ !  
! +00000000+ +00000000+!  
! +00000000+ +00000000+!  
! +00000000+ +00000000+!  
! +X00000X+ +X00000X+ !  
!  
! +X00000X+ +X00000X+ !  
! +00000000+ +00000000+!
```

Figure D-3. Spatial Derivative in y

tensity function. Within the computer simulation, subroutine Deriv implements this algorithm. This appendix derives the spatial derivative of an intensity function. As shown in Chapter IV the algorithm will require the negative of this result.

Appendix E

Generation of White Gaussian Noise Process

Throughout this research, it was desired to generate samples of a discrete-time white Gaussian noise vector process with a mean of zero and a given variance. This function must be simulated through the use of pseudorandom codes that generate numbers as though they were generated as samples of a scalar random variable with uniform probability density between 0 and 1. (see Figure E-1)

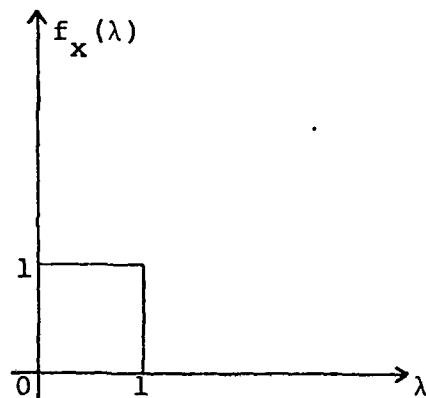


Figure E-1. Output Probability Density of Pseudorandom Code

The mean and variance of this distribution is given by

$$\mu_\lambda = \int_{-\infty}^{\infty} \lambda f_x(\lambda) d\lambda = \int_0^1 \lambda d\lambda = \frac{1}{2} \quad (E-1)$$

$$\sigma_\lambda^2 = \int_{-\infty}^{\infty} (\lambda - \frac{1}{2})^2 f_x(\lambda) d\lambda = \int_0^1 [\lambda^2 - \lambda + \frac{1}{4}] d\lambda = \frac{1}{12} \quad (E-2)$$

If twelve independent calls are made to such a random number

generator and the outputs of these calls, specific realizations ξ_i , are summed, as

$$\Omega_j = \sum_{i=1}^{12} \xi_i \quad (E-3)$$

the result is a realization of a random variable with a mean of 6 and a variance of 1 that is essentially Gaussian (by the Central Limit Theorem, Law of Large Numbers, etc.). Six is then subtracted from each sum of realizations, Ω_j , to produce a discrete realization of ξ_j which will have a Gaussian distribution of zero mean and unity variance

$$\xi_j = \Omega_j - 6 \quad (E-4)$$

Now the ξ_j 's are arrayed into a vector of appropriate dimension. For example to create a 64×1 vector \underline{w}_1 , 64 realizations of ξ_j , each being created from independent calls to the pseudorandom code, would be arrayed:

$$\underline{w}_1 = \begin{bmatrix} \xi_1 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \xi_{64} \end{bmatrix} \quad (E-5)$$

The vector process \underline{w}_1 is composed of independent scalar white Gaussian noises of mean zero and unit variance.

To create a discrete-time white Gaussian noise vector process $\underline{w}_d(t_i)$, described by mean of zero and covariance of $\underline{\Omega}_d(t_i)$, in general non-diagonal, the following equation can be used.

$$\underline{w}_d(t_i) = \sqrt{\underline{\Omega}_d(t_i)} \underline{w}_l(t_i) \quad (E-6)$$

where $\sqrt{\cdot}$ denotes Cholesky lower triangular square root. (13:408)

Reference 13 shows that Equation (E-6) properly models the desired characteristics.

In summary, this appendix shows how samples of discrete-time white Gaussian noise vectors were generated from pseudo-random computer codes.

Appendix F

Discrete Representation of the Derivative of the Intensity Profile with Respect to the Kalman Filter States

To present a clearer presentation of the discrete representation of the intensity profile and its derivative with respect to the Kalman filter states, the continuous one-dimensional Gaussian profile of Figure F-1a is used. If the discrete representation of this intensity function is approximated as a single value at the center of a pixel the Figure F-1b is appropriate.

At time t_1 , assume that the intensity profile is centered in the field-of-view of eight horizontal pixels, as in Figure F-1b. Although h is actually the averaged value over a pixel it is approximated as the sampled value at the center. At time t_2 , the target has moved in the direction of increasing state by an amount x_0 , see Figure F-2. Taking the first pixel as an example, h_1 decreased in magnitude as the state increased. The magnitudes of the intensity measurements for all pixels located at state values to the left of the profile's maximum decreased in intensity while those to the right increased in magnitude. This is the spatial derivative rotated 180 degrees about the vertical axis. Figure F-3 gives the spatial derivative of a one-dimensional intensity profile as in Figure F-1, while Figure F-4 gives the derivative with respect to the Kalman filter states.

In summary, the derivative of the intensity profile with respect to the Kalman filter states is the negative of the spatial derivative of that intensity profile with respect to coordinate directions.

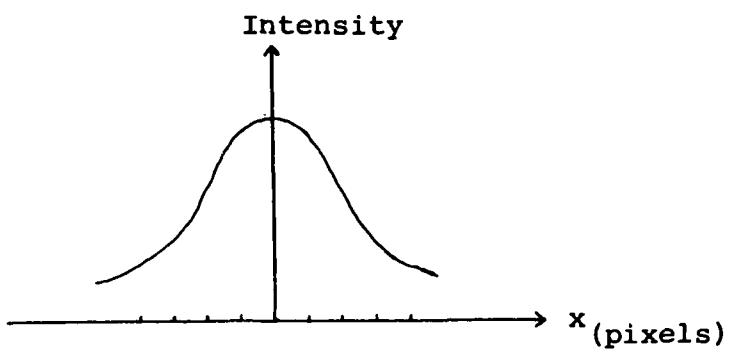


Figure F-1a. Continuous One-Dimensional Gaussian

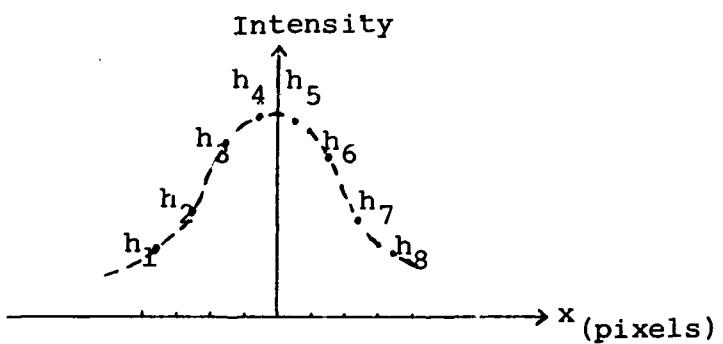


Figure F-1b. Discrete One-Dimensional Gaussian

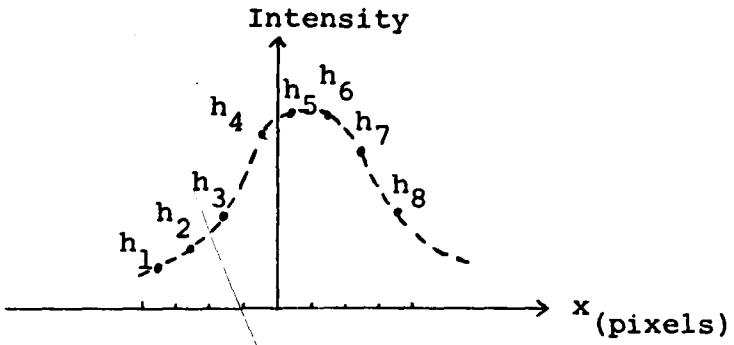


Figure F-2. Shifted Intensity Profile

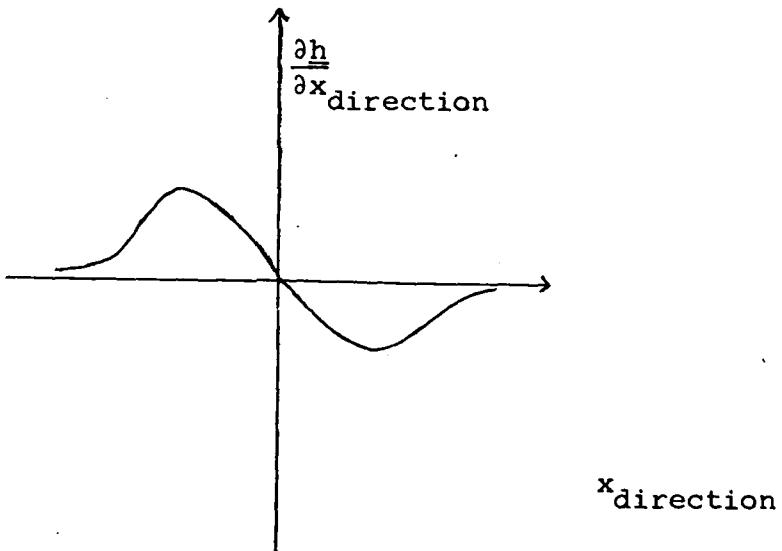


Figure F-3. Intensity's Spatial Derivative

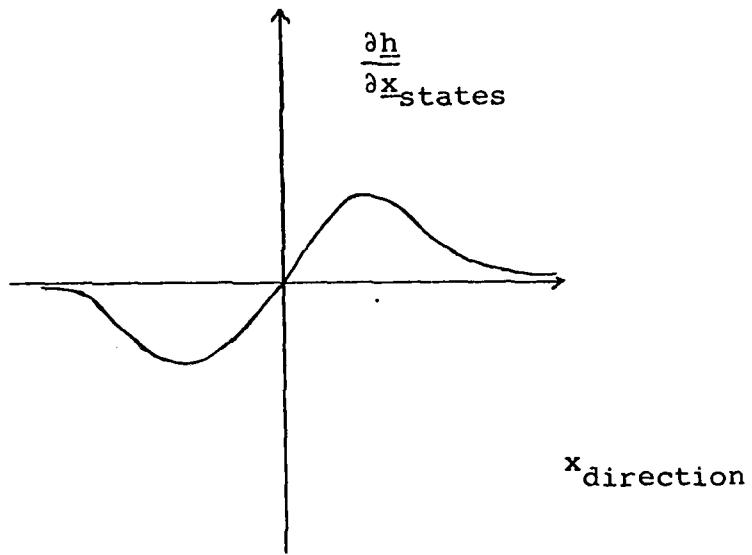


Figure F-4. Derivative with Respect to States

Appendix G

Monte Carlo Study

For the Monte Carlo analysis, many samples of the error process are generated within the computer simulation, and the error statistics are computed from those samples. In this research, twenty FLIR data frames represent one tracking history in time. For each quantity of interest, errors are computed before and after processing a FLIR data frame. These critical values, which the Kalman filter estimates, are compared to the truth model output to compute errors, as depicted in Figure G-1 (13:327).

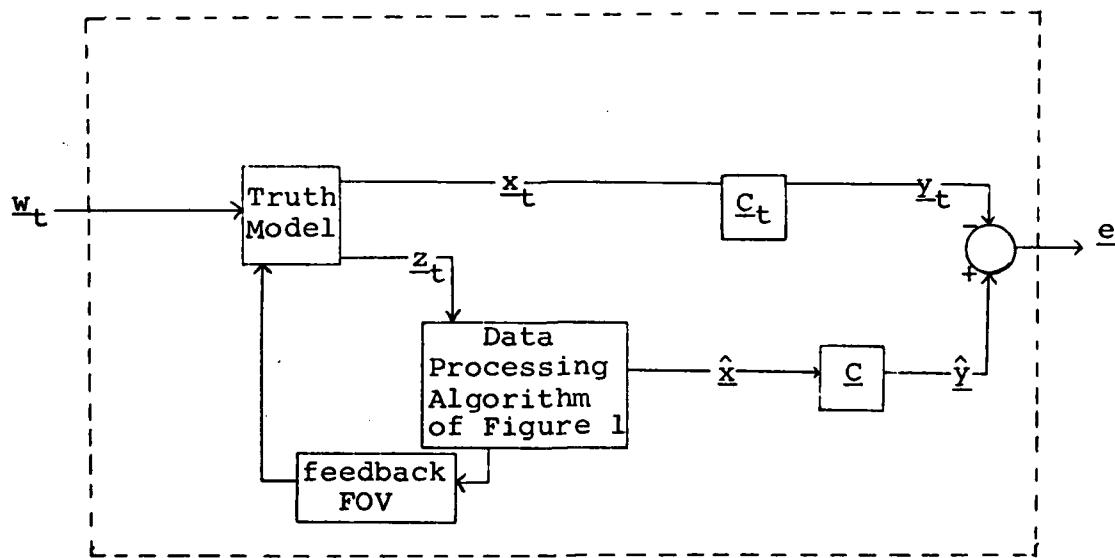


Figure G-1. Generation of Error Samples

The truth model generates the measurement vector $\underline{z}_t(t_i)$ which is a realization of the measurement process $\underline{z}_t(\cdot, \cdot)$ at time t_i , i.e. $\underline{z}_t(t_i, \omega_k)$. The feedback loop only controls the center of the field-of-view for which the intensity measurements are generated. Twenty such twenty-frame time histories are now processed and the statistics of the errors, at each time frame are computed by averaging across the twenty time histories. For example say the difference between the truth model value for x dynamics, x_{td} , and the filter estimated value for x dynamics, \hat{x}_d , in the first time history, at frame one was e_{11x_d} . By keeping stored e_{ikx_d} and $e_{ikx_d}^2$ for each frame i in a given simulation run, $i=1, \dots, 20$, and for the twenty time histories $k=1, 2, \dots, 20$, the mean error and the variance of the errors for any frame can be computed.

Figure G-1 assumes that the true values of the critical quantities y_t are related to the truth model states, \underline{x}_t , by a linear transformation represented by the matrix C_t . The matrix C is similarly explained. For every quantity of critical interest, the appropriate entries in the 'C' matrices designate what combination of states constitute that value so the error sample can be generated.

In summary, the object of this Monte Carlo study was to characterize the error process of the algorithm of Figure 1 statistically.

Appendix H

Computer Software

This appendix contains the Fortran source code for some of the computer programs written in this study. The first program contained in this appendix was written for use on the CDC Fortran IV compiler. This program is the completed implementation of the algorithm of Figure 1. The next program was written for use on the MODCOMP Classic minicomputer to implement the algorithm of Chapter V. The last listing is the program which was written to generate the plots of Chapter VI. This software is well commented to allow for ease of understanding and modification.

PROGRAM MAIN 74/74 OPT=1 PHDM3 FTN 4.8+528 09/21/81 12.15.51 PAGE 1
 1 PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE8=
 1 DEBUG=OUTPUT)
 C*****
 C*
 C* COMMENTS ON DATA STRUCTURES
 C*
 C*****
 C*
 C* C*** IMAX IS AN ARRAY WHICH CONTAINS PEAK VALUES FOR THE THREE GAUSSIANS
 C* C*** RMS BACKGROUND DIVIDED INTO THIS MAX VALUE QUAL SNR
 C*
 C* S IS AN ARRAY WHICH CONTAINS INVERSE COVARIANCES OF STATES
 C*
 C* XMAX IS A REAL ARRAY WHICH CONTAINS LOCATION OF MAX X COODOF 3 GAUSS
 C*
 C* YMAX IS SIMILAR TO XMAX
 C*
 C* R IS THE CORRELATION MATRIX FILLED BY SUBROUTINE SPTN
 C* IT CONTAINS THE SPATIAL NOISE CORRELATION COEFFICIENT
 C*
 C* REAL IMAX(3),S(12),XMAX(3),YMAX(3),R(64,64)
 C*
 C* XTREAL A/P OF TRUTH MODEL STATES
 C* SEE SUB TRUTH
 C*
 C* PHIT- TRUTH MODEL STATE TRANSITION MATRIXSEE SUB TRUTH
 C* QD- QDOT-CHOLEKY SQUARE ROOT OF QD=E(WND)
 C* VT- VLT-WHAT WILL QD BY 10 CORRUPT THE TRUTH MODEL STATES PG 15 MERCIER
 C*
 C* H- THE MATRIX WHICH DETERMINES THE OUTPUT COMBINATION OF STATES
 C* YT- OUTPUT EQUATIONS
 C* REAL XT(3,1),PHIT(3,3),QDRDQT(8,8),WT(4,1),H(2,8),YT(2,1)
 C* ERGOT- ROOT OF R
 C*
 C* Y- VECTOR OF INDEP WHIUTE GAUSS N(0,1)
 C* WILL MULTI BY RJECT TO GET STATE UNCERTAINTY NOIS CORRUPTION
 C*
 C* V-RROOT*W
 C*
 C* UC-FOR OUTSIDE TRACKING WINDOW NEED SPATIALLY UNCORRELATED 24X24
 C* REAL P(X0(1,64),W(04),V(64),UC(576)
 C*
 C* DATE- ERROR IN RECONSTRUCTION OF DATA ARRAY. ONE NUMBER PER PIXELPER FRAME
 C*
 C* DXE- ERROR IN FORMATION OF DERIVATIVE WITH RESPECT TO X
 C*
 C* DYE- SEE DXE
 C*
 C* REAL DATE1(8,20),DATE(8,8,20),DYE(8,8,20)
 C* DATE1- SOLVE(CF,E1,L1)---->SMALL1(8,20) DXE2
 C* REAL DATE2(8,8,20),DXE2(8,8,20),DATE2(E,E,20)
 C*
 C* NV IS AN ARRAY WHICH CONTAINS DIMENSION OF DATA SEE FOURT

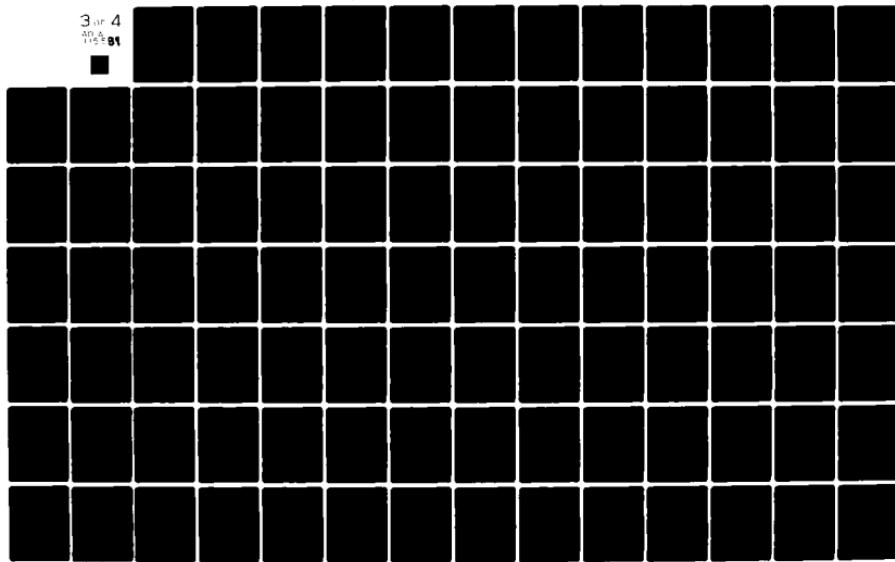
AD-A115 581 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHO0--ETC F/G 12/1
ENHANCED TRACKING OF AIRBORNE TARGETS USING FORWARD LOOKING INF--ETC(U)
DEC 81 S K ROGERS

UNCLASSIFIED

AFIT/GEO/EE/81D-5

NL

3 or 4
AD-A
115 581



PROGRAM MAIN 74/74 OPT=1 PNDMP F7N 4.8+528 09/21/81 12.15.51 PAGE 2

```

C** INTEGER NN(2)
C** DATA- COMPLEX ARRAY OF DATA TO USED BY FOURT
C** WORK- SEE COMMENTS IN FOURT
C** DX , DY,-DERIVATIVES
C** COMPLEX DATA(24,24),WORK(50),DX(24,24),DY(24,24)
C** SD-PERFECT SHIFT/MANIPULATED DATA-SAVE DATA
C** SX,SY-SAVE PERFECT D/DX,D/DY
C** SDATA- SMOOTHED DATA
C** COMPLEX SD(24,24),SX(24,24),SY(24,24),SDATA(24,24)

C FILTERS DATA STRUCTURES

C
C PHIF IS THE STATE TRANSITION MATRIX FOR THE KALMAN FILTER
C -SEE SUBROUTINE FILTER
C QFD IS THE RESULT OF THE INTEGRAL TERM IN THE PROPAGATION
C -OF THE COV MATRIX SEE SUBROUTINE PROPF
C PFP IS THE FILTER'S COVARIANCE MATRIX PLUS- AFTER INCORPORATION
C -OF A MEASUREMENT
C PFM IS THE FILTERS COVARIANCE MATRIX MINUS AFTER PROPAGATION
C -BUT PRIOR TO MEASUREMENT INCORPORATION
C XFP IS THE FILTER STATE VECTOR PLUS
C XFM IS THE FILTER STATE VECTOR MINUS

C REAL PHIF(4,4),QFD(4,4),PFP(4,4),PFM(4,4),XFP(4),XFM(4)

C RINV IS THE INVERSE OF R WHICH IS NEEDED FOR THE PROPAGATION
C - OF THE INVERSE COVARIANCE METHOD
C REAL RINV(64,64)

C Z IS THE KALMAN FILTER MEASUREMENT VECTOR
C REAL Z(64)

C LINH IS AN ARRAY WHICH IS NEEDED IN THE EXTENDED KALMAN FILTER
C -THE PARTIAL OF THE INTENSITY FUNCTION WITH RESPECT TO
C -FILTERS BEST ESTIMATE OF STATES
C NLINH IS AN ARRAY WHICH IS FILLED FROM THE SMOOTHED DATA AND IS
C -THE NONLINEAR INTENSITY FUNCTION WHICH IS TO BE USED TO
C -PROCESS THE NEXT MEASURE*CNT
C REAL LINH(64,4),NLINH(64)

C SAVE IS A TEMPORARY USE MATRIX IN CONJUNCTION WITH SDATA
C COMPLEX SAVE(24,24)
C

```


PROGRAM MAIN 74/74 OPT=1 PMDMP FTN 4.8+528 09/21/81 12.15.51 PAGE 4

C IFLAG WILL DETERMINE IF ANOTHER SET OF VARIATIONS TO THE INPUT
C NEEDS TO BE PROCESSED
C

175 C C SETUP IS THE INITIALIZATION ROUTINE
C CALL SETUP(COV,S,NZ,NZM,NFRAMES,NRUNS,ALPHA,NFREQ,
11SF,IEF,VARM,SIGDOT,DT,TD,IFLAG,PHIT,QDROOT,H,TAF,
2VARDF,VARAF,TDF,PHIF,QFD,DATE,DATE2,DXE,DXE2,
3DYE,DYE2,XFME,XFME2,CNME,CNME2,XFPE,XFPE2,
4CNPE,CNPE2,IMAX,XMAX,YMAX,X,Y,SD,SX,SY,R,RROOT,RINV)
IF(IFLAG.NE.0) GO TO 6421
C C

C*****
C*** BEGIN MONTE CARLO SIMULATION LOOP ***
C*****
C C

180 C C
190 C C
195 C C
200 C C
205 C C
175 C C
210 C C
215 C C
220 C C
225 C C

DO 90 NS=1,NRUNS
XSHIFT=0.0
YSHIFT=0.0
C INITIALIZE SMOOTH DATA
C
DO 7 I=1,24
DO 7 J=1,24
SDATA(I,J)=CMPLX(0.,0.)
C
ZERO INITIAL CONDITIONS FOR THE FILTER
C
DO 106 I=1,4
DO 106 J=1,4
PFP(I,J)=0.0
PFP(1,J)=0.0
XFP(I)=0.0
XFM(I)=0.0
PFP(1,1)=0.0375
PFP(2,2)=.0375
PFP(3,3)=.0025
PFP(4,4)=.0025
C
INITIAL CONDITIONS ON DYNAMIC STATES
XFP(1)=3.0
XFP(2)=3.0
XFM(1)=XFP(1)
XFM(2)=XFP(2)
C
INITIALIZE STATE VECTOR
C
DO 71 I=1,8

PROGRAM	MAIN	74/74	OPT=1	PMDMP	FTN 4.8+528	09/21/81	12.15.51	PAGE	5
230	C	71	XT(1,1)=0.						
	C		C YT(1,1) IS THE XPEAK OF THE CENTROID						
	C		C YT(2,1) IS THE YPEAK OF THE CENTROID						
	XT(1,1)=3.0								
235	XT(2,1)=3.0								
	YT(1,1)=3.0								
	YT(2,1)=3.0								
	C		C DEFINE UPPER LEFT CORNER OF FOV						
240	C		C CENTERS CENTROID IN THE FOV						
	X=XFP(1)-4.								
	Y=XFP(2)-4.								
245	C		C TRACK TARGET FOR NFRAME FRAMES (TIME SLICES)						
	C		DO 90 NR=1,NFRAMES						
	C		C DEFINE GAUS PEAK LOCATIONS BASED ON CENTROID POSITION,YT						
250	C		C XMAX(1)=YT(1,1)						
	C		C XMAX(2)=YT(1,1)-2.						
	C		C XMAX(3)=YT(1,1)+2.						
	C		C YMAY(1)=YT(2,1)-2.666666						
	C		C YMAY(2)=YT(2,1)+1.333333						
	C		C YMAY(3)=YT(2,1)+1.333333						
	C		C WRITE(6,410)						
	C		C FORMAT(//,* X TRUTH*,4X,* YTRUTH*,4X,* X FOV*,4X,* Y FOV*)						
	C		C WRITE(6,411) YT(1,1),YT(2,1),X,Y						
	C		C FORMAT(//,4E10.3)						
	C		C WRITE(6,174) NR,YT(1,1),YT(2,1),X,Y,XFM(1),XFM(2),XT(1,1),XT(2,1)						
255	C		C FORMAT(T2,* FRAME :* T12,T12,* XPOS :*,E10.3,T27,* YPOS :*,E10.3,T42,						
	C		C /*FOV :*,E10.3,T57,*YFOV :*,E10.3,*T12,*XFM :*,E12.5,T30,*YFM :*,						
	C		C E12.5,/,T12,*XT :*,E12.5,T30,*YT :*,E12.5)						
260	C		C GET MEASUREMENT NOISE ARRAY						
	C		C						
	C		C CALL NOISE(W,64)						
	C		C CALL MULT(RROOT,W,64,64,1,V)						
	C		C GET MEASUREMENT DATA						
	C		C						
275	C		C CALL INPUT3(I1MAX,S,XMAX,YMAX,24,X,Y,DATA,CENX,CENY)						
	C		C CALL IDEAL(I1MAX,S,XMAX,YMAX,24,NZ,X,Y,DATA,DX,DY)						
	C		C IF(NR.EQ.1) CALL DISPLAY(24,24,DATA)						
	C		C IF(NR.EQ.1) CALL DISPLAY(24,24,DX)						
	C		C IF(NR.EQ.1) CALL DISPLAY(24,24,DY)						
280	C		C ADD CORRELATED MEASUREMENT NOISE AT CENTER ,X8						
	C		C						
295	C		DO 4 I=1,8						

PROGRAM	MAIN	74/74	OPT=1	PMDMP	FTN 4.8+528	09/21/81	12.15.51	PAGE	6
290	C	DO 4 J=1,B DATA(I+B,J+B)=DATA(I+B,J+B)+CMPLX(V(B*(I-1)+J),0,0) CONTINUE							
	C	ADD UNCORRELATED NOISE TO MEASUREMENT DATA OUTSIDE CENTER							
	C	CALL NOISE(LUC,576)							
	DO 6 I=1,24								
	DO 6 J=1,24								
295	I IF(I.GE.9.AND.I.LE.16.AND.J.GE.9.AND.J.LE.16) GO TO 6 IF((I.LE.NZ).OR.(J.LE.NZ).OR.(I.GE.NZM).OR.(J.GE.NZM)) GO TO 6 IF((I.LE.NZ).OR.(J.LE.NZ).OR.(I.GE.NZM).OR.(J.GE.NZM)) GO TO 6 DATA(I,J)=DATA(I,J)+CMPLX(UC(24*(I-1)+J),0.)*VARM CONTINUE								
300	C								
	C	GET FORWARD FFT							
	C								
305	C	CREATE THE MEASUREMENT VECTOR FOR THE FILTER UPDATE							
	C								
	C	K=0							
	DO 101 I=9,16								
	DO 101 J=9,16								
	K=K+1								
	Z(K)=REAL(DATA(I,J))								
	C								
310	C	101							
	C								
315	C	GO CALCULATE THE ERRORS OF THE FILTERS ESTIMATE PRIOR TO MEASUREMENT INCORPORATION							
	C								
	C	CALL STATE(XFME,XFME2,CNME,CNME2,XFM,XT,YT,NR)							
320	C								
	C	INCORPORATE MEASUREMENT							
	C								
	C	IF(NR.EQ.1) GO TO 164							
325	C	CALL UPDAT(Z,LINH,XFM,PFP,PFM,RINV)							
	C	CALCULATE THE ERRORS FOR THE FILTER AFTER THE INCORPORATION OF THE MEASUREMENT							
	C								
330	C	CALL STATP(XFPE,XFPE2,CNPE,CNPE2,XT,YT,XFP,NR)							
	C	COMPUTE THE SHIFT INFORMATION FROM THE CENTER OF FOV							
	C	XSHIFT=X-XFP(1)+4.-XFP(3)							
	C	YSHIFT=Y-XFP(2)+4.-XFP(4)							
	C	WRITE(6,173) XSHIFT,YSHIFT							
335	C	FORMAT(T12,*XSHIFT:*,E10.3,T42,*YSIFT:*,E10.3)							
	C								
	C								
340	C	164 CALL FOURT(DATA,NN,2,-1,1,WORK)							
	C								
	C	FILTER DESIRED FREQUENCY							

PROGRAM MAIN 74/74 OPT=1 PMDMP

FTN 4.8+529 09/21/81 12.15.51 PAGE 7

```
C
C
C      IF(NFREQ.GT.12) NFREQ=12
C      IF(NFREQ.LE.0) GO TO 3796
C      FROM START FREQ TO END FREQ
DO 8   I=1SF,IEF
DO 8   J=1,24
DATA(I,J)=CMPLX(0.,0.)
DATA(J,I)=CMPLX(0.,0.)
CONTINUE
C
C
355   C      IF(NR.NE.1) CALL SHIFT(DATA,24,XSHIFT,YSHIFT)
C      CALL SMOOTH(DATA,SDATA,ALPHA,24,NR)
C
C      CALL DISPLAY(24,24,SX)
C      CALL DISPLAY(24,24,SDATA)
C      CALL DISPLAY(24,24,DX)
C      CALL DISPLAY(24,24,SY)
C      CALL DISPLAY(24,24,DY)
C
C      CALL PROPF(PHIF,QFD,PFP,PFM,XFP,XFM)
X=XFM(1)-4.
Y=XFM(2)-4.
XSHIFT=XFM(3)
YSHIFT=XFM(4)
370   C      ROUTINE FOR COMPUTING PERFECT FOR ERROR COMPUTATION
C      CALL PERF(X,Y,XSHIFT,YSHIFT,SD,SX,SY,IMAX,XMAX,YMAX,S,NZ)
C
C      IF DESIRE PERFECT INTENSITY FUNCTION DATA UNCOMMENT NEXT CALL TO
C      TO PERF AND COMMENT 14 LINES FOLLOWING IT
C      CALL PERF(X,Y,XSHIFT,YSHIFT,SAVE,DY,IMAX,XMAX,YMAX,S,NZ)
DO 170 I=1,24
DO 170 J=1,24
SAVE(I,J)=SDATA(I,J)
CALL SHIFT(SAVE,24,XSHIFT,YSHIFT)
CALL DERIV(SAVE,24,DX,DY)
CALL FOURT(SAVE,NN,2,1,1,WORK)
CALL FOURT(DX,NN,2,1,1,WORK)
CALL FOURT(DY,NN,2,1,1,WORK)
385   C      SCALE THE INVERSE TRANSFORM ALONG WITH SETTING THE APPROPRIATE
C      SIGN TO INDICATE THE CHANGE IN INTENSITY PATTERN WITH RESPECT TO A
C      CHANGE IN STATE
DO 172 I=1,24
DO 172 J=1,24
DX(I,J)=-DX(I,J)/576.
SAVE(I,J)=SAVE(I,J)/576.
DY(I,J)=-DY(I,J)/576.
172
C      FILL IN THE LINEARIZED INTENSITY ARRAY
C
395   C      K=0
C      DO 102 I=9,16
```

PROGRAM MAIN	74/74	OPT=1	PMDMP	FTN 4.8+528	09/21/81	12.15.51	PAGE
400	DO 102 J=9,16						8
	K=K+1						
	LINH(K,1)=REAL(DX(I,J))						
	LINH(K,3)=LINH(K,1)						
	LINH(K,2)=REAL(DY(I,J))						
405	NLINH(K)=REAL(SAVE(I,J))						
	LINH(K,4)=LINH(K,2)						
	C ACCUMULATE ERRORS IN INTENSITY FUNCTIONS						
	C CALL ACCUM(SUMDATA,SAVE,SD,SUMDX,DX,SX,SUMDY,DY,SY,						
	#DATE,DATE2,DXE,DXE2,DYE,DYE2,NR)						
410	C PROPAGATE TRUTH MODEL STATE ONE FRAME						
	C CALL PROP(PHIT,QDROOT,H,XT,YT,B,2)						
	C						
	90 CONTINUE						
	C*** END MONTE CARLO SIMULATION LOOP ****						
	C*** #NFRAMES ****						
415	C						
	C CALCULATE MEAN AND VARIANCE STATISTICS						
	C						
	420 C GO CALCULATE THE FILTER STATISTICS						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	425 C STOP						
	C END						
	C						
	430 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	435 C STOP						
	C END						
	C						
	440 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	445 C STOP						
	C END						
	C						
	450 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	455 C STOP						
	C END						
	C						
	460 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	465 C STOP						
	C END						
	C						
	470 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	475 C STOP						
	C END						
	C						
	480 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	485 C STOP						
	C END						
	C						
	490 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	495 C STOP						
	C END						
	C						
	500 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	505 C STOP						
	C END						
	C						
	510 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	515 C STOP						
	C END						
	C						
	520 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	525 C STOP						
	C END						
	C						
	530 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	535 C STOP						
	C END						
	C						
	540 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	545 C STOP						
	C END						
	C						
	550 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	555 C STOP						
	C END						
	C						
	560 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	565 C STOP						
	C END						
	C						
	570 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	575 C STOP						
	C END						
	C						
	580 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	585 C STOP						
	C END						
	C						
	590 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	595 C STOP						
	C END						
	C						
	600 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	605 C STOP						
	C END						
	C						
	610 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	615 C STOP						
	C END						
	C						
	620 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	625 C STOP						
	C END						
	C						
	630 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	635 C STOP						
	C END						
	C						
	640 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						
	C*** #VARM,ALPHA,XSHIFT,YSHIFT,SIGDT ****						
	C GO TO 3985						
	645 C STOP						
	C END						
	C						
	650 C						
	C CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NRUNS,						
	C*** #NFRAMES ****						
	C						
	C CALL STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DATE2,DXE2,DYE2,NZ,NFREQ,COV,						

SYMBOLIC REFERENCE MAP (R=1)

VARIABLES	SN	TYPE	RELOCATION	74/74	OPT=1	PMDMP
36077	DYE	REAL	ARRAY	45477	DYE2	REAL
17555	H	REAL	ARRAY	7305	I	INTEGER
7272	IEF	INTEGER	ARRAY	7261	IFLAG	INTEGER
7314	IMAX	REAL	ARRAY	7271	ISF	INTEGER
7306	J	INTEGER	ARRAY	7310	K	INTEGER
100255	LINH	REAL	ARRAY	7265	NFRAMES	INTEGER
7270	NFREQ	INTEGER	ARRAY	100655	NLINH	REAL
50077	NN	INTEGER	ARRAY	7307	NR	INTEGER
7266	NRUNS	INTEGER	INTEGER	7302	NS	INTEGER
7263	NZ	INTEGER	REAL	7264	NZM	INTEGER
70125	PFM	REAL	ARRAY	70105	PFP	REAL
70045	PHIF	REAL	ARRAY	17351	PHIT	REAL
17451	QDROOT	REAL	ARRAY	70065	QFD	REAL
7341	R	REAL	ARRAY	70155	RINV	REAL
17577	RROOT	REAL	ARRAY	73117	S	REAL
100755	SAVE	COMPLEX	ARRAY	57045	SD	COMPLEX
65545	SDATA	COMPLEX	ARRAY	7274	SIGDT	REAL
	SUMDATA	REAL	REAL	7312	SUMDX	REAL
7313	SUNDY	REAL	COMPLEX	61245	SX	COMPLEX
63445	SY	REAL	ARRAY	7277	TAF	REAL
7276	TD	REAL	ARRAY	7200	TDF	REAL
27777	UC	REAL	ARRAY	27677	V	REAL
7301	VARAF	REAL	REAL	7300	VARDF	REAL
7273	VARM	REAL	COMPLEX	27577	W	REAL
52301	WORK	REAL	ARRAY	17551	WT	REAL
7176	X	REAL	ARRAY	70151	XFM	REAL
103155	XFME	REAL	ARRAY	103225	XFME2	REAL
70145	XF2	REAL	ARRAY	103415	XFPE	REAL
103465	XFPE2	REAL	ARRAY	7333	XMAX	REAL
7303	XSHIFT	REAL	ARRAY	17341	XT	REAL
7177	Y	REAL	REAL	73336	YMAX	REAL
7304	YSHIFT	REAL	REAL	17575	YT	REAL
100155	Z	REAL	ARRAY			

FILE NAMES	MODE	INPUT	OUTPUT	C TAPES
2054	DEBUG	0	TAPE8	2054
2054	TAPE8	4130	INPUT	

EXTERNALS	TYPE	ARGS	DERIV	FOURT	MULT	PERF	PROPF	SHIFT	STAT	STATFP
ACCUM	16	10	4	6	6	12	6	4	16	8
FILST	10	11								
IDEAL	11	2								
NOISE	2	7								
PROP	7	49								
SETUP	5	5								
SMOOTH	8	8								
STATFM										
UPDAT										

INLINE FUNCTIONS	TYPE	ARGS	REAL	REAL	1 INTRIN
CMPXL	COMPLEX	2 INTRIN			

STATEMENT LABELS	TYPE	ARGS	REAL	REAL	1 INTRIN
0	4	6377	6	0	7
0	8	0	71	0	90
0	101	0	102	0	106
6442	164	0	170	0	172

PROGRAM MAIN 74/74 OPT=1 PWDMP FTN 4.8+528 09/21/81 12.15.51 PAGE 10

STATEMENT LABELS FMT NO REFS FMT NO REFS
7234 173 FMT NO REFS
7210 411 FMT NO REFS
6631 6421

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES	EXT REFS	NOT INNER
6225	90	NS	194 418	3778			
6230	7	1	200 202	14B	INSTACK	NOT INNER	
6236	7	J	201 202	2B	INSTACK	NOT INNER	
6245	106	I	208 213	14B	INSTACK		
6251	106	J	209 213	4B	INSTACK		
6272	71	I	228 229	2B	INSTACK		
6305	90	NR	247 418	315B	EXT REFS	NOT INNER	
6324	4	I	285 288	178	INSTACK	NOT INNER	
6334	4	J	286 288	3B	INSTACK		
6345	6	I	293 298	37B	OPT	NOT INNER	
6364	6	J	294 298	15B	OPT	NOT INNER	
6406	101	I	309 312	15B	INSTACK	NOT INNER	
6414	101	J	310 312	3B	INSTACK	NOT INNER	
6453	8	I	348 351	17B	INSTACK	NOT INNER	
6463	8	J	349 351	3B	INSTACK		
6513	170	I	378 380	163	INSTACK	NOT INNER	
6522	170	J	379 380	3B	INSTACK		
6543	172	I	389 393	26B	OPT	NOT INNER	
6553	172	J	390 393	13B	INSTACK	NOT INNER	
6573	102	I	399 406	21B			
6601	102	J	400 406	7B			

STATISTICS
PROGRAM LENGTH 76037B 31775
BUFFER LENGTH 52000B CM USED 5616B 2958

SUBROUTINE SETUP

FTN 4-8+528 09/21/81 12. 15. 51 PAGE 2

09/21/81 12.15,51

09/21/81 12:15:51 PAGE 2

```

C
C          DO 9 I=1,B
C          DO 9 J=1,8
C          DO 9 K=1,20
C          DATE(I,J,K)=0.
C          DATE2(I,J,K)=0.
C          DXE(I,J,K)=0.
C          DYE(I,J,K)=0.
C          DXE2(I,J,K)=0.
C          DYER(I,J,K)=0.
C
C          9      C      INITIALIZE THE FILTER ERROR MATRICES TO ZERO
C
C          DO 21 I=1,2
C          DO 21 J=1,20
C          XFM(E(I,J))=0.
C          XFM(E2(I,J))=0.
C          CNM(E(I,J))=0.
C          CNM(E2(I,J))=0.
C          CNPE(E(I,J))=0.
C          CNPE(E2(I,J))=0.
C          CNPE2(E(I,J))=0.
C          CNPE2(E2(I,J))=0.
C
C          21     C      SET UP IDEAL DATA FOR ERROR CALCULATION
C
C          CALL IDEAL(IMAX,S,XMAX,YMAX,24,NZ,X,Y,SD,SX,SY)
C
C          USING FIRST AND SECOND NEAREST NEIGHBOR DETERMINED
C          SQUAREROOT , RROT, OF THE MEAS COVARIANCE MATRIX,
C
C
C          CALL SPNTN(VARM,R,8)
C          LOOP TO MAKE SPATIALLY UNCORRELATED IF NEXT 4
C          DO 6428 I=1,64
C          DO 6428 J=1,64
C          R(I,J)=0.0
C          IF(I.EQ.J) R(I,J)=VARM
C          CONTINUE
C          6428   CALL CHOLY(R,64,RROOT)
C
C          GET THE INVERSE OF THE CORRELATION COEFFICIENT
C          - NEEDED FOR THE INVERSE COV METHOD
C          CALL INVERT(R,64,RINV)
C
C          END INITIALIZATION
C
C          RETURN
C
C          6421  IFLAG=1

```


SUBROUTINE SETUP 74/74 OPT=1 PMDMP FTN 4.8+528 09/21/81 12.15.51 PAGE 4

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
112	9	K	62 68	4B	INSTACK
125	21	I	73 82	17B	NOT INNER
133	21	J	74 82	5B	INSTACK

STATISTICS
PROGRAM LENGTH 3528 234
52000B CM USED

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 ACCUM

SUBROUTINE ACCUM			74/74	OPT=1	PMDMP	FTN 4.8+528	09/21/81	12.15.51	PAGE
LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES				2
7	2	I	13 27	41B	NOT INNER				
17	2	J	14 27	25B	OPT				
STATISTICS									
PROGRAM LENGTH			74B	60					
52000B CM USED									

```

1      SUBROUTINE PERF 74/74  OPT=1 PMDMP   FTN 4.8+528  09/21/81 12.15.51
2
3      SUBROUTINE PERF(X,Y,XSHIFT,YSHIFT,SD,SX,SY,IMAX,YMAX,S,NZ)
4      COMPLEX SD(24,24),SX(24,24),SY(24,24)
5      REAL IMAX(3),YMAX(3),S(12)
6
7      C THIS ROUTINE COMPUTES THE POERFECT DATA WITHIN A 24X24 ARRAY SD
8      C WHICH IS OFFSET FROM THE CENTER OF THE FOV BY XSHIFT AND
9      C YSHIFT WITHOUT ANY TRANSFORMS IT ALSO COMPUTES THE DERIVATIVE
10     C
11     C
12     XMAX( 1)=X+4.+XSHIFT
13     XMAX( 2)=X+2.+XSHIFT
14     XMAX( 3)=X+6.+XSHIFT
15     YMAX( 1)=Y+4.+YSHIFT-2.6666666
16     YMAX( 2)=Y+4.+YSHIFT+1.3333333
17     YMAX( 3)=Y+4.+YSHIFT+1.3333333
18     CALL IDEAL(IMAX,S,XMAX,YMAX,24,NZ,X,Y,SD,SX,SY)
19     DO 1 I=1,24
20     DO 1 J=1,24
21     SX(I,J)=-SX(I,J)
22     SY(I,J)=-SY(I,J)
23     RETURN
24

```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

VARIABLES	SN	TYPE	INITIAL	RELLOCATION	0	IMAX	REAL	ARRAY
107	1	INTEGER			0	NZ	INTEGER	ARRAY
110	J	INTEGER			0	SD	COMPLEX	ARRAY
		REAL			0	SY	COMPLEX	ARRAY
0	SX	CHAR(1)X	ARRAY	F.P.	0	XMAX	REAL	ARRAY
0	X	REAL	ARRAY	F.P.	0	Y	REAL	ARRAY
0	XSHIFT	REAL	ARRAY	F.P.	0	YSHIFT	REAL	ARRAY
0	YMAX	REAL	ARRAY	F.P.				
EXTERNALS								
IDEAL								
STATEMENT LABELS								
0	1							
LOOPS	LABEL	INDEX		FROM-TO	LENGTH	PROPERTIES	NOT INNER	
4.4	1	I		16 19	209	INSTACK		
54	1	J		17 19	58			
STATISTICS	PROGRAM LENGTH	CODE LENGTH	CACHE USED	115B	77			

PAGE

09/21/81 12.15.51

SUBROUTINE STATFM 74/74 OPT=1 PMOMP

FTN 4.8+528

```

1      SUBROUTINE STATFM(XFME,XFME2,CNME,CNME2,XFM,XT,YT,NR)
      REAL XFME(2,20),XFME2(2,20),CNME(2,20),CNME2(2,20)
      REAL XFM(4),XT(8,1),YT(2,1)
```

```

5      C THIS ROUTINE GATHERS THE INFORMATION THAT WILL BE
      C REQUIRED TO COMPUTE THE STATISTICS OF THE PREDICTIONS
      C OF THE FILTER PRIOR TO MEASUREMENT INCORPORATION
```

```

10     C FIRST COLLECT THE ERROR IN THE PREDICTED DYNAMIC LOCATION
```

```

C XFME(1,NR)=XFME(1,NR)+XFM(1)-XT(1,1)
C XFME(2,NR)=XFME(2,NR)+XFM(2)-XT(2,1)
```

```

C NOW COLLECT THE SQUARE OF THAT ERROR
```

```

15     C XFME2(1,NR)=XFME2(1,NR)+(XFME(1,NR)-XT(1,1))**2
      C XFME2(2,NR)=XFME2(2,NR)+(XFME(2,NR)-XT(2,1))**2
```

```

C COLLECT ERROR IN CENTROID PREDICTED LOCATION MINUS
```

```

20     C CNME(1,NR)=CNME(1,NR)+(XFM(1)+XFME(3)-YT(1,1))
      C CNME(2,NR)=CNME(2,NR)+(XFM(2)+XFME(4)-YT(2,1))
```

```

C COLLECT THE SQUARE OF THE ERROR
```

```

25     C CNME2(1,NR)=CNME2(1,NR)+(XFME(1)+XFM(3)-YT(1,1))**2
      C CNME2(2,NR)=CNME2(2,NR)+(XFME(2)+XFM(4)-YT(2,1))**2
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 STATFM

VARIABLES	SN	TYPE	RELOCATION	0	CNME2	REAL	ARRAY	F.P.
0 CNME		REAL	ARRAY	0	XFM	REAL	ARRAY	F.P.
0 NR		INTEGER		0	XFME2	REAL	ARRAY	F.P.
0 XFM		REAL	ARRAY	0	YT	REAL	ARRAY	F.P.
0 XT		REAL	ARRAY					

STATISTICS
PROGRAM LENGTH
52000B CM USED

43B 35

```

1      SUBROUTINE STATFP(XFPE,XFPE2,CNPE,CNPE2,XT,YT,XFP,NR)
      REAL XFPE(2,20),XFPE2(2,20),CNPE(2,20),CNPE2(2,20)
      REAL XT(8,1),YT(2,1),XFP(4)

5      C THIS ROUTINE GATHERS THE INFORMATION THAT WILL BE
      C REQUIRED TO COMPUTE THE STATISTICS ON THE FILTERS
      C UPDATED STATE ESTIMATES

10     C COMPUTE DIFFERENCES THAT WILL BE NEEDED
      C
      C DIF1=XFP(1)-XT(1,1)
      C DIF2=XFP(2)-XT(2,1)
      C DIF3=XFP(1)+XFP(3)-YT(1,1)
      C DIF4=XFP(2)+XFP(4)-YT(2,1)

15     C FIRST COLLECT THE ERROR IN THE DYNAMIC LOCATION ESTIMATES
      C
      C XFPE(1,NR)=XFPE(1,NR)+DIF1
      C XFPE(2,NR)=XFPE(2,NR)+DIF2
      C NOW COLLECT THE SQUARE OF THAT ERROR
      C
      C XFPE2(1,NR)=XFPE2(1,NR)+DIF1**2
      C XFPE2(2,NR)=XFPE2(2,NR)+DIF2**2
      C NOW COLLECT THE ERROR IN THE CENTROID UPDATE
      C
      C CNPE(1,NR)=CNPE(1,NR)+DIF3
      C CNPE(2,NR)=CNPE(2,NR)+DIF4
      C NOW COLLECT THE ERROR SQUARED
      C
      C CNPE2(1,NR)=CNPE2(1,NR)+DIF3**2
      C CNPE2(2,NR)=CNPE2(2,NR)+DIF4**2
      C RETURN
      C END

```

190

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 STATFP

VARIABLES	SN	TYPE	ARRAY	RELOCATION		
0 CNPE		REAL		F.P.	CNPE2	REAL
44 DIF1		REAL			DIF2	REAL
46 DIF3		REAL			DIF4	REAL
0 NR		INTEGER		F.P.	XFP	REAL
0 XFPE		REAL	ARRAY	F.P.	XFPE2	REAL
0 XT		REAL	ARRAY	F.P.	YT	REAL
						ARRAY

SUBROUTINE	STATFP	74/74	OPT=1	PMDMP			
STATISTICS					FTN 4.8+528	09/21/81	12.15.51
PROGRAM LENGTH	52000B	CM USED	50B	40			PAGE
							2

```

1      SUBROUTINE FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,
#NRUNS,NFRAMES)
REAL XFME(2,20),XFME2(2,20),XFPE(2,20),XFPE2(2,20)
REAL CNME(2,20),CNME2(2,20),CNPE(2,20),CNPE2(2,20)

5      C THIS ROUTINE COMPUTES THE STATISTICS ON THE FILTER ERRORS

C
DO 1 I=1,2
DO 1 J=1,NFRAMES
XFME(I,J)=XFME(I,J)/FLOAT(NRUNS)
XFPE(I,J)=XFPE(I,J)/FLOAT(NRUNS)
CNME(I,J)=CNME(I,J)/FLOAT(NRUNS)
CNPE(I,J)=CNPE(I,J)/FLOAT(NRUNS)
XFME2(I,J)=SORT(ABS(XFME2(I,J))/FLOAT(NRUNS)-XFME(I,J)**2)
XFPE2(I,J)=SQRT(ABS(XFPE2(I,J))/FLOAT(NRUNS)-XFPE(I,J)**2)
CNME2(I,J)=SQRT(ABS(CNME2(I,J))/FLOAT(NRUNS)-CNME(I,J)**2)
CNPE2(I,J)=SQRT(ABS(CNPE2(I,J))/FLOAT(NRUNS)-CNPE(I,J)**2)
WRITE(6,2)
FORMAT(T2,*FRAME*,T10,*XERR(-)*,T30,*SXERR(-)*,T55,*YERR(-)*,
20      *T68,*SYERR(-)*)
DO 3 I=1,NFRAMES
WRITE(6,4) I,XFME(1,I),XFME2(1,I),XFME(2,I
#),XFME2(2,I)
FORMAT(T4,12,T9,E12.5,T28,E12.5,T52,E12.5,T66,E12.5)
CONTINUE
4      3
FORMAT(T2,*FRAME*,T10,*XERR(+)*,T30,*SXERR(+)*,T55,*YERR(+)*,
5      *T68,*SYERR(+)*)
DO 6 I=1,NFRAMES
WRITE(6,7) I,XFPE(1,I),XFPE2(1,I),XFPE(2,I),XFPE2(2,I)
FORMAT(T4,12,T9,E12.5,T28,E12.5,T52,E12.5,T66,E12.5)
CONTINUE
6      WRITE(6,101)
FORMAT(T2,*FRAME*,T10,*CNER(-)*,T30,*SCNER(-)*,T55,*YCER(-)*,
35      *T68,*SYCR(-)*)
DO 102 I=1,NFRAMES
WRITE(6,103) I,CNME(1,I),CNME2(1,I),CNME(2,I),CNME2(2,I)
FORMAT(T4,12,T9,E12.5,T28,E12.5,T52,E12.5,T66,E12.5)
CONTINUE
102   WRITE(6,10)
FORMAT(T2,*FRAME*,T10,*CNER(+)*,T30,*SCNER(+)*,T55,*YCER(+)*,
10      *T68,*SYCR(+)*)
DO 103 I=1,NFRAMES
WRITE(6,9) I,CNPE(1,I),CNPE2(1,I),CNPE(2,I),CNPE2(2,I)
FORMAT(T4,12,T9,E12.5,T28,E12.5,T52,E12.5,T66,E12.5)
CONTINUE
45      9
8      RETURN
END

```

SUBROUTINE FILST 74/74 OPT=1 PMDMP

FTN 4.8+528

09/21/81 12.15.51

PAGE 2

ENTRY POINTS

3 FILST

VARIABLES	SN	TYPE	RELOCATION				
0 CNME		REAL	ARRAY F.P.	0 CNME2	REAL	ARRAY	F.P.
0 CNPE		REAL	ARRAY F.P.	0 CNPE2	REAL	ARRAY	F.P.
317 1		INTEGER		320 J	INTEGER		
0 NFRAMES		INTEGER		0 NRUNS	INTEGER		
0 XFME		REAL	ARRAY F.P.	0 XFME2	REAL	ARRAY	F.P.
0 XFPE		REAL	ARRAY F.P.	0 XFPE2	REAL	ARRAY	F.P.
FILE NAMES		MODE					
		TAPE6					
EXTERNALS		TYPE					
SQRT		REAL					
INLINE FUNCTIONS		TYPE					
ABS		REAL					
STATEMENT LABELS							
0 1				152 2	FMT		0 3
173 4		FMT		204 5	FMT		0 6
225 7		FMT		0 8			311 9
270 10		FMT		236 101	FMT		0 102
257 103		FMT					
LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES	EXT REFS	NOT INNER
7 1	1		8 17	473		EXT REFS	
10 1	1	J	9 17	448		EXT REFS	
61 3	1		21 25	123		EXT REFS	
77 6	1		29 32	148		EXT REFS	
115 102	1		36 39	148		EXT REFS	
133 8	1		43 46	148		EXT REFS	
STATISTICS	PROGRAM LENGTH						
	5200008 CM USED			373B	251		

	SUBROUTINE INITF	74/74	OPT*1	PNDMP	FTN 4.8+52B	09/21/81	12.15.51	PAGE
1	C	SUBROUTINE INITF(TAF,VARDF,VARAF)						1
	C	THIS ROUTINE CONTROLS INPUTING VALUES NEEDED FOR THE KALMAN						
5	C	C - FILTER						
	C							
	C	TAF=.07072						
10	C	THIS IS THE CORRELATION TIME FOR THE ATMOSPHERIC MODEL FOR THE						
	C	- FILTER						
	C							
	C	VARDF=1.0						
15	C	THE VARIANCE OF THE ATMOSPHERIC JITTER FOR THE FILTER						
	C							
	C	VARAF=0.001						
	C	RETURN						
	C	END						
20								

SYMBOLIC REFERENCE MAP (R=1)

194	ENTRY POINTS							
	3	INITF						
	VARIABLES	SN	TYPE	RELOCATION				
	0 TAF		REAL	F.P.				
	0 VARDF		REAL	F.P.				
	STATISTICS				O VARAF	REAL		F.P.
	PROGRAM LENGTH							
	52000B CM USED				15B	13		

SUBROUTINE FILTER 74/74 OPT=1 PMDMF FTN 4.8+528 09/21/81 12.15.51 PAGE 1

```

1   SUBROUTINE FILTER(TDF,VARDF,TAF,VARAF,DT,PHIF,QFD)
      REAL PHIF(4,4),QFD(4,4)

5   C THIS ROUTINE SETS UP THE STATE TRANSITION MATRIX AND QFD MATRIX
      C
      C TAF CORRELATION TIME FOR THE ATMOSPHERIC JITTER
      C TDF CORRELATION TIME FOR THE TARGET DYNAMICS
      C VARDF TARGET DYNAMICS NOISE VARIANCE
      C VARAF ATMOSPHERIC NOISE VARIANCE
      C D(XF(T))/DT-TIME DERIVATIVE OF FILTER STATE
      C FILFT-FILTER PLANT MATRIX
      C WFT INPUT WHITE NOISE VECTOR FOR FILTER
      C STATE SPACE MODEL
      C

10   C
      C D(XF(T))/DT=FILFT*XF(T)+WFT(T)

15   C WHERE
      C
      C FILFT = : -1/TDF    0    0    0    :
      C        : 0    -1/TDF    0    0    :
      C        : 0    0    -1/TAF    0    :
      C        : 0    0    0    -1/TAF :
      C
      C E\WFT(T)=0

20   C
      C WHERE
      C
      C QF = : 2*VARDF/TDF    0    0    0    :
      C        : 0    2*VARDF/TDF    0    0    :
      C        : 0    0    2*VARAF/TAF    0    :
      C        : 0    0    0    2*VARAF/TAF :
      C
      C THE SOLUTION TO THE DYNAMIC EQUATIONS
      C
      C XF(I+1)=PHIF*XF(I)

25   C FOR PROPAGATION OF COVARIANCE MATRIX NEED QFD
      C
      C DO 1 I=1,4
      C     DO 1 J=1,4
      C         PHIF(I,J)=0.
      C         QFD(I,J)=0.
      C
      C         PHIF(1,1)=EXP(-DT/TDF)
      C         PHIF(2,2)=PHIF(1,1)
      C         PHIF(3,3)=EXP(-DT/TDF)
      C         PHIF(4,4)=PHIF(3,3)
      C
      C         QFD(1,1)=TDF*(1.-EXP(-2.*DT/TDF))
      C         QFD(2,2)=QFD(1,1)
      C         QFD(3,3)=VARAF*(1.-EXP(-2.*DT/TAF))
      C
      C         QFD(4,4)=QFD(3,3)
      C
      C RETURN
      C
      C END
  
```

SUBROUTINE FILTER 74/74 OPT=1 PMOMP

FTN 4.8+528 09/21/81 12.15.51 PAGE 2

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 FILTER

VARIABLES	SN	TYPE	RELOCATION
0 DT		REAL	F.P.
53 J		INTEGER	
0 QFD		REAL	ARRAY F.P.
0 TDF		REAL	F.P.
0 VAROF		REAL	F.P.

EXTERNALS	TYPE	ARGS	LIBRARY
EXP	REAL	1	

STATEMENT LABELS

0	1
---	---

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
7	1	I	39 42	14B	NOT INNER
15	1	J	40 42	3B	INSTACK

STATISTICS	PROGRAM LENGTH	CM USED	60B	48
	52000B			

SUBROUTINE PROPF 74/74 OPT=1 PMDMP FTN 4.84528 09/21/81 12.15.51 PAGE 1

```

1      SUBROUTINE PROPF(PHIF,QFD,PFP,PFM,XFM)
      REAL PHIF(4,4),QFD(4,4),PFP(4,4),PFM(4,4),XFP(4),XFM(4)
      REAL TEMP1(4,4),TEMP2(4,4)

5      C THIS ROUTINE IMPLEMENTS THE STATE TRANSITION
      C -EQUATIONS FOR THE FILTER
      C
      C XF(I+1)=PHIF*XF(I)

10     C PM=PIFI*PFP+PHIFT +QFD
      C WHERE
      C
      C PHIF=FILTER STATE TRANSITION MATRIX
      C XF =FILTER STATE VECTOR
      C PFM =COV FILTER STATES MINUS
      C PFP =COV FILTER STATES PLUS

15     C
      C WHERE
      C QFD = : VARDF*(1.-EXP(-2.*DT/TOF)) 0 0 0 :
      C          : 0 VARDF*(1.-EXP(-2.*DT/TDF)) 0 0 0 :
      C          : 0 0 VARAF*(1.-EXP(-2.*DT/TAF)) 0 0 0 :
      C          : 0 0 0 VARAF*(1.-EXP(-2.*DT/TAFF)) 0 0 0 :

20     C
      C PERFORM FILTER STATE PROPAGATION

25     C CALL MULT(PHIF,XFP,4,4,1,XFM)
      C ADD DETERMINISTIC INPUT OF 3 PIXELS PER TIME HISTORY IN THE X DIR
      C XFM(1)=XFM(1)+15
      C CALL MULT(PHIF,PFP,4,4,TEMP1)
      C CALL MULT(TEMP1,PHIF,4,4,TEMP2)
      C DO 1 I=1,4
      C DO 1 J=1,4
      C PFM(I,J)=TEMP2(I,J)+QFD(I,J)
      C RETURN
      C END

30     1
      C
      C SYMBOLIC REFERENCE MAP (R=1)

      ENTRY POINTS
      3 PROPF
      VARIABLES
      75  I   SN  TYPE           RELOCATION
      0   PFM  INTEGER          76  J   INTEGER
      0   PHIF  REAL             0   PFP  REAL
      0   TEMP1 REAL             0   QFD  REAL
      77  TEMP1 REAL             117 TEMP2  REAL
      0   XFM  REAL             0   XFP  REAL
      EXTERNALS
      MULT  TYPE           ARG'S
      6
      STATEMENT LABELS
      1
  
```

SUBROUTINE 'ROPF			74/74 OPT=1 PMDMP			FTN 4.8+528			09/21/81 12.15.51			PAGE
LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES							2
27	1	1	30 32	168	NOT INNER							
36	1	J	31 32	38	INSTACK							
STATISTICS												
PROGRAM LENGTH	143B	99										
S2000B CM USED												

SUBROUTINE	UPDAT	74/74	OPT=1	PMDMP	FTN 4.8+528	09/21/81	12.15.51	PAGE
1								1
5	C	SUBROUTINE UPDAT(Z,LINH,NLINH,XFP,XFM,PFP,PFM,RINV)						
	C	REAL Z(64), LINH(64,4),NLINH(64,4),LINHT(4,64),XFP(4,4)						
	C	REAL PFP(4,4), PFM(4,4),RINV(64,64),TEMP1(4,64),XFM(4,						
	C	REAL RESID(64), PINV(4,4),GAINK(4,64),TEMP2(4,4)						
10	C	C THIS ROUTINE PROCESSES ONE MEASUREMENT VECTOR Z						
	C	AND ALSO UPDATES THE FILTER STATES XFP						
	C	Z IS THE MEASUREMENT VECTOR OF FLIR MEASUREMENTS						
	C	NLINH IS THE NONLINEAR INTENSITY FUNCTION AFTER SMOOTHING						
15	C	LINH IS THE LINEAR INTENSITY FUNCTION						
	C	XFP IS FILTER STATES PLUS						
	C	XFM IS FILTER STATES MINUS						
	C	PFP COV MATRIX OF FILTER STATES PLUS						
	C	PFM COV MATRIX OF FILTER STATES MINUS						
	C	RINV INVERSE OF SPATIAL CORRELATION COEF MATRIX						
	C	TEMP1 WILL HOLD H-TRANSPOSE*RINV						
	C	PINV WILL HOLD COV MATRIX MINUS INVERSED						
	C	TEMP2 WILL HOLD H*INVH						
	C	GAINK KALMAN FILTER GAIN						
20	C	RESID RESIDUAL						
	C	UPD GAIN TIMES RESIDUAL						
	C	THE EQUATIONS USED FOR THE UPDATE ARE						
	C	FIRST FOR CG,PUTINE THE COVARIANCE PLUS MATRIX						
25	C	USING THE INVERSE COV METHOD						
	C	PINV(I+)=PINV(I-)+LINHT(I)*RINV(I)*LINH(I)						
	C	PFP(I+)=PINV(I+)-!						
30	C	K(I)+P(I+)*LINHT(I)*RINV(I)						
	C	XFP(I)=XFM(I-)+K(I)*Z(I)-NLINH(I)!						
35	C	FIRST CREAT LINH TRANSPOSED						
	C	DO 1 I=1,64						
	C	DO 1 J=1,4						
	C	LINHT(J,I)=LINH(I,J)						
40	C	1 COMPUTE THE INVERSE OF THE COV PLUS MATRIX						
	C	CALL MULT(LINH,RINV,4,64,64,TEMP1)						
	C	CALL MULT(TEMP1,LINH,4,64,4,TEMP2)						
45	C	C INVERT THE COVARIANCE MINUS MATRIX						
	C	CALL INVERT(PFM,4,PINV)						
50	C	DO 2 I=1,4						
	C	DO 2 J=1,4						
	C	PINV(I,J)=PINV(I,J)+TEMP2(I,J)						
	C	CREATE P(I+)-PFP						
	C	C CALL INVERT(PINV,4,PFP)						
	C	C COMPUTE THE INVERSE OF THE COV PLUS MATRIX						
	C	C CALL MULT(LINH,RINV,4,64,64,TEMP1)						
	C	C CALL MULT(TEMP1,LINH,4,64,4,TEMP2)						

SUBROUTINE UPDAT

FTN 4.8+528

09/21/81

12.15.51

PAGE 2

```

C
C COMPUTE STATE MEASUREMENT UPDATE
DO 3 I=1,64
  RESID(I)=Z(I)-NLINH(I)
  CALL MULT(GAINK,RESID,4,64,1,UPD)
DO 4 I=1,4
  XFP(I)=XFM(I)+UPD(I)
RETURN
END

```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3 UPDAT

VARIABLES

	SN	TYPE	ARRAY	RELOCATION		
1312	GAINK	REAL	ARRAY		150	I
151	J	INTEGER	ARRAY		0	LINH
152	LINHT	REAL	ARRAY		0	NLINH
0	PFM	REAL	ARRAY	F.P.	0	PFM
1272	PINV	REAL	ARRAY		1172	RESID
0	RINV	REAL	ARRAY	F.P.	552	TEMP1
1152	TEMP2	REAL	ARRAY	F.P.	1712	UPD
0	XFM	REAL	ARRAY	F.P.	0	XFP
0	Z	REAL	ARRAY	F.P.		

EXTERNALS

INVERT TYPE ARGS 3

MULT

STATEMENT LABELS

0 2

0 6

0 3

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
7	1	I	37 39	14B	NOT INNER
15	1	J	39 39	3B	INSTACK
25	2	I	49 51	14B	NOT INNER
42	2	J	50 51	3B	INSTACK
63	3	I	60 61	3B	INSTACK
75	4	I	63 64	3B	INSTACK

STATISTICS
PROGRAM LENGTH 52000B CM USED

1720B 976

SUBROUTINE STAT 74/74 OPT=1 FMDMP

FTN 4.8+528 09/21/81 12.15.51 PAGE 1

```
1      SUBROUTINE STAT(NFRAMES,DATE,NRUNS,DYE,DXE,DYE2,DXE2,NZ,
  #NFRQ,COV,VARM,ALPHA,XSHIFT,YSHIFT,SIGDT)
  #REAL DATE(8,8,20),DXE(8,8,20),DYE(8,8,20),DATE2(8,8,20),
  #DXE2(8,8,20),DYE2(8,8,20)
  DO 10 K=1,NFRAMES
  DO 10 J=1,8
  DO 10 I=1,8
  DATE(I,J,K)=DATE(I,J,K)/NRUNS
  DYE(I,J,K)=DYE(I,J,K)/NRUNS
  DYE(I,J,K)=DYE(I,J,K)/NRUNS
  DATE2(I,J,K)=DATE2(I,J,K)/NRUNS-DATE(I,J,K)**2
  DXE2(I,J,K)=DXE2(I,J,K)/NRUNS-DXE(I,J,K)**2
  DYE2(I,J,K)=DYE2(I,J,K)/NRUNS-DYE(I,J,K)**2
  CONTINUE
  10     WRITE(6,9987) NRUNS,NFRAMES,NZ,NFRQ,COV,VARM,ALPHA,
  #SIGDT
  9987    FORMAT(1H1,110,*RUNS=*,I2,T40,*FRAMES=*,I2,T70,*NUMBER ZERO PAGES*,
  #   I1,T100,*NUMBER FREQ ZEROED=*,I2,/T10,*GAUSSIAN COVARIANCE*,
  #   F5.2,T40,*BACKGROUND VARIANCE=*,F5.1,T70,*SMOOTHING ALPHA=*,,
  #F7.3./,
  / T70.*TRUTH MODEL UNCERTAINTY=*,F5.2,///)
  DO 11 K=1,NFRAMES
  C      WRITE(6,9988) ((DATE(I,J,K),J=1,8),I=1,8)
  C      WRITE(6,9988) ((DXE(I,J,K),J=1,8),I=1,8)
  C      WRITE(6,9988) ((DYE(I,J,K),J=1,8),I=1,8)
  C      WRITE(6,9988) ((DATE2(I,J,K),J=1,8),I=1,8)
  C      WRITE(6,9988) ((DXE2(I,J,K),J=1,8),I=1,8)
  C      WRITE(6,9988) ((DYE2(I,J,K),J=1,8),I=1,8)
  C      WRITE(6,9988) ((DXE2(I,J,K),J=1,8),I=1,8)
  C      CONTINUE
  11     9988    FORMAT(//,(2X,BE16.5))
  9324    WRITE(6,9324)
  20      FORMAT(T2,*FRAME*,T15,*MEAN ERROR*,T33,*MEAN VARIANCE*,T54,
  #*MEAN VARIANCE ,T71,*MEAN VARIANCE OF*,T94,*MEAN ERROR*,T111,
  #*MEAN VARIANCE OF*,/T32,*OF ERROR IN H*,T56,*IN D/DX*,
  #T71,* ERROR IN D/DX*,T96,*IN D/DY*,T111,* ERROR IN D/DY*)
  DO 13 K=1,NFRAMES
  SUMH=0.
  VARH=0.
  SUMDX=0.
  VARDX=0.
  SUMDY=0.
  VARDY=0.
  DO 12 I=1,8
  DO 12 J=1,8
  SUMH=SUMH+DATE(I,J,K)/64.
  VAPH=VARH+DATE2(I,J,K)/64.
  SUMDX=SUMDX+DXE(I,J,K)/64.
  VARDX=VARDX+DXE2(I,J,K)/64.
  SUMDY=SUMDY+DYE(I,J,K)/64.
  VARDY=VARDY+DYE2(I,J,K)/64.
  CONTINUE
  12     WRITE(6,9325) K,SLWH,VARH,SUMDX,SUMDY,VARDY
  9325    FORMAT(T4,I2,T13,E12.5,T33,E12.5,T53,E12.5,T73,E12.5,T93,
  #CONTINUE
  13     RETURN
  END
```

SUBROUTINE STAT

FTN 4.8+528

09/21/81 12.15.51 PAGE 2

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

74/74 DPT=1 PMDMP

FTN 4.8+528

09/21/81 12.15.51 PAGE 2

VARIABLES	SN	TYPE	RELOCATION	REAL	COV	REAL	ARRAY	F.P.
0 ALPHA		REAL	F.P.		0	DATE2	REAL	F.P.
0 DATE		REAL	F.P.		0	DXE2	REAL	F.P.
0 DXE		REAL	ARRAY		0	DYE2	REAL	F.P.
0 DYE		REAL	ARRAY		250	J	INTEGER	F.P.
247 I		INTEGER			0	NFRAMES	INTEGER	F.P.
246 K		INTEGER			0	NRUNS	INTEGER	F.P.
0 NREQ		INTEGER			0	SIGDT	REAL	F.P.
0 NZ		INTEGER			0	SUNDY	REAL	F.P.
253 SUMDX		REAL			254	VARDX	REAL	F.P.
251 SUMH		REAL			252	VARH	REAL	F.P.
256 VARDY		REAL			0	XSHIFT	REAL	F.P.
0 VARM		REAL					*UNUSED	F.P.
0 YSHIFT		REAL	*UNUSED					
FILE NAMES		MODE						
TAPE6		FMT						
STATEMENT LABELS								
0 10				0	11	9324	FMT	
0 13				171				
132 9987		FMT		163	9988	FMT	NO REFS	
202 LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES	NOT INNER		
7 10	K		5 14	32B		NOT INNER		
10 10	I		6 14	27B		NOT INNER		
21 10	J		7 14	12B	OPT			
16 11	K		22 29	1B	INSTACK			
53 13	K		36 55	43B	EXT REFS	NOT INNER		
60 12	I		43 51	31B		NOT INNER		
67 12	J		44 51	17B	OPT			
STATISTICS								
PROGRAM LENGTH								
52000B CM USED				300B	192			

SUBROUTINE INIT 74/74 OPT=1 PMOMP FTN 4.8+528 09/21/81 12.15.51 PAGE 1
 1 SUBROUTINE INIT(COV,S,NZ,NZM,NFRAMES,NRUNS,ALPHA,NFREQ,ISF,
 #IEF,VARM,SIGDT,DT,TD)
 DIMENSION S(12);
 C
 C
 C DEFINE TRUE TARGET AS 3 INDEP GAUSS FUNCTNSA WITH VAR=COV
 C
 C
 10 S(1)=1./COV
 S(4)=S(1)
 S(5)=S(1)
 S(8)=S(1)
 S(9)=S(1)
 S(12)=S(1)
 WRITE(6,3791)
 FORMAT(1X,*NUMBER OF ZEROES TO PAD*)
 READ(5,561) NZ
 FORMAT(1I2)
 NZm=25;NZ
 WRITE(6,3792)
 FORMAT(1X,*NUMBER OF FRAMES*)
 READ(5,561) NFRAMES
 WRITE(6,4023)
 FORMAT(1X,*NUMBER OF SIMULATIONS*)
 READ(5,561) NRUNS
 WRITE(6,3793)
 FORMAT(1X,*ALPHA FOR SMOOTHING*)
 READ(5,560) ALPHA
 WRITE(6,3795)
 FORMAT(1X,*NUMBER OF HIGH FREQ COMPONENTS TO ZERO*)
 READ(5,561) NFREQ
 ISF=4-NFREQ
 IEF=12+NFREQ
 WRITE(6,3789)
 FORMAT(1X,*INPUT BACKGROUND VARIANCE*)
 READ(5,560) VARM
 560 FORMAT(1F6.2)
 WRITE(6,965)
 FORMAT(2X,*VARIANCE OF TRUTH MODEL DYNAMICS UNCERTAINTY*)
 READ(5,410) SIGDT
 FORMAT(1F6.2)
 DT=.0333333
 TD=1.
 RETURN
 END

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 INIT

SUBROUTINE INIT 74/74 OPT=1 PMDMP

PAGE 2

FTN 4.8+528

09/21/81 12.15.51

VARIABLES	SN	TYPE	RELOCATION			
0 ALPHA		REAL	F.P.	0 COV	REAL	F.P.
0 DT		REAL	F.P.	0 IEF	INTEGER	F.P.
0 ISF		INTEGER	F.P.	0 NFRAMES	INTEGER	F.P.
0 NFREQ		INTEGER	F.P.	0 NRUNS	INTEGER	F.P.
0 NZ		INTEGER	F.P.	0 NZM	INTEGER	F.P.
0 S		REAL	ARRAY F.P.	0 SIGOT	REAL	F.P.
0 TD		REAL	F.P.	0 VARM	REAL	F.P.

FILE NAMES	MODE	TAPE6	FMT			
STATEMENT LABELS	FMT					
217 410	FMT	176	560	FMT	75	561
203 965	FMT	164	3789	FMT	63	3791
102 3792	FMT	132	3793	FMT	0	3794
146 3795	FMT	116	4023	FMT		

STATISTICS	PROGRAM LENGTH	223B	147			
	52000B CM USED					

SUBROUTINE TRUTH 74/74 OPT=1 PMDMMP FTN 4.8+528 09/21/81 12.15.51 PAGE 1

```
1        SUBROUTINE TRUTH(PHIT,QDROOT,H,SIGDT,DT,TD)
      REAL PHIT(8,8),QD(8,8),K,QDROOT(8,8),H(2,8)
      REAL QDPAR(6,6)
      REAL QDPRT(6,6)

5        C    ASSUME TARGET DYNAMICS ARE: XD=WD/(S+1/TD)
      C         YD=WD/(S+1/TD)
      C         XA=WA*K*A*B**2/(S+A)/(S+B)**2
      C         YA=XA
```

```
10       C    THEN AN EQUIVALENT STATE SPACE MODEL IS:
      C         D(XD)/DT=FT*XD+GT*WT    AND    YD=H*XD
      C    WHERE
```

```
15       C         : -1/TD    0    0    0    0    0    0    0    :
      C         : 0    -1/TD    0    0    0    0    0    0    :
      C         : 0    0    -A    0    0    0    0    0    :
      C         : 0    0    0    -B    1    0    0    0    :
      C         : 0    0    0    0    0    -B    0    0    :
      C         : 0    0    0    0    0    0    -A    0    :
      C         : 0    0    0    0    0    0    0    -B    :
      C         : 0    0    0    0    0    0    0    1    :
      C         : 0    0    0    0    0    0    0    0    -B    :

20       C         FT=    : 0    0    0    0    0    0    0    0    :
      C         : 0    0    0    0    0    0    0    0    :
      C         : 0    0    0    0    0    0    0    0    :
      C         : 0    0    0    0    0    0    0    0    :
      C         : 0    0    0    0    0    0    0    0    :
      C         : 0    0    0    0    0    0    0    0    :
      C         : 0    0    0    0    0    0    0    0    :
      C         : 0    0    0    0    0    0    0    0    :
      C         : 0    0    0    0    0    0    0    0    :

25       C         GT=    : 1    0    0    0    0    0    0    0    :
      C         : 0    1    0    0    0    0    0    0    :
      C         : 0    0    G1    0    0    0    0    0    :
      C         : 0    0    0    G2    0    0    0    0    :
      C         : 0    0    0    0    G3    0    0    0    :
      C         : 0    0    0    0    0    G1    0    0    :
      C         : 0    0    0    0    0    0    G2    0    :
      C         : 0    0    0    0    0    0    0    G3    0    :

30       C         AND    HT=    : 1    0    1    0    0    0    0    0    :
      C         : 0    1    0    0    0    0    0    0    :
      C         : 0    0    G1    0    0    0    0    0    :
      C         : 0    0    0    G2    0    0    0    0    :
      C         : 0    0    0    0    G3    0    0    0    :
      C         : 0    0    0    0    0    H1    0    0    :
      C         : 0    0    0    0    0    0    H2    0    :
      C         : 0    0    0    0    0    0    0    H3    0    :

35       C         : 0    0    0    0    0    0    0    0    :
```

```
40       C         XD=    : XT    :    :
      C         : YT    :    :
      C         : X1A :    :
      C         : X2A :    :
      C         : X3A :    :
      C         : Y1A :    :
      C         : Y2A :    :
      C         : Y3A :    :
```

50 C THE SOLUTION OF THE DYNAMIC EQUATIONS IS

```
55       C         XD(I+1)= PHIT*XD(I) + SQRT(QD)*WT
      C    WHERE PHIT=STATE TRANSITION MATRIX
      C    SIGDT= DYNAMIC NOISE STANDARD DEVIATION
      C    WT= GAUSSIAN NOISE VECTOR
```

FTN 4.8+528 09/21/81 12.15.51 PAGE 2

SUBROUTINE	TRUTH	74/74	.OPT=1	PMDMP
60	C	C	QD= COVARIANCE MATRIX	
65	C	NSTATE=8 K=.3882109544*SIGDT A=14.14 B=659.5		
70	C	DO 1 I=1,NSTATE DO 1 J=1,NSTATE PHIT(I,J)=0. 1 QD(I,J)=0.	PHIT(1,1)=EXP(-DT/TD) PHIT(2,2)=EXP(-DT/TD)	THESE NEXT TWO CARDS CHANGE THE TRUTH MODEL TO A DETERMINISTIC CONSTANT VELOCITY MODEL OF 3 PIXELS PER TIME HISTORY IN THE X D
75	C	PHIT(1,1)=1. PHIT(2,2)=PHIT(1,1) PHIT(3,3)=EXP(-A*DT)	PHIT(4,4)=EXP(-B*DT) PHIT(4,5)=DT*EXP(-3*DT)	
80	C	PHIT(5,5)=EXP(-B*DT) PHIT(6,6)=EXP(-A*DT) PHIT(7,7)=EXP(-B*DT) PHIT(7,8)=-DT*EXP(-B*DT)	PHIT(8,8)=EXP(-B*DT)	
85	C	FACT1=(K*2)*(A**2)*(B**..) FACT2=A+B FACT3=2.*B	FACT1=FACT1/FACT1**4 G1=FACT1/(FACT1**3) G2=FACT1/(FACT1**2) G3=FACT1/(FACT1**2) P1=1.-EXP(-2.*A*DT) P2=1.-EXP(-FACT2*DT) P3=1.-EXP(-2.*B*DT) P4=DT*EXP(-FACT2*DT) P5=DT*EXP(-2.*B*DT)	BY SETTING THESE ELEMENTS OF QD =0 ONLY A PERFECT CONSTANT VEL
90		QD(1,1)=SIGDT**2*(1.-EXP(-2.*DT/TD))*TD/2.	P1=1.-EXP(-2.*A*DT) P2=1.-EXP(-FACT2*DT) P3=1.-EXP(-2.*B*DT) P4=DT*EXP(-FACT2*DT) P5=DT*EXP(-2.*B*DT)	
95	C	QD1(1,1)=0.0 CD(2,2)=QD(1,1) QD(3,3)=(G1*P1)/(2.*A)	QD(3,4)=P2*(G2/FACT2**2-G1/FACT12)-P4*G2/FACT2	
100		CD(3,5)=G2*P2/FACT2 QD(4,3)=QD(3,4)	QD(4,4)=P3*(G1/FACT3**2+2.*G2/FACT2**2-G3/FACT3**3)- P5*(-G2/B+G3*DT/FACT3**2.*G3/FACT3**2)	
105		CD(4,5)=P3*(G3/FACT3**2-G2/FACT13)-P5*G3/FACT3 QD(5,3)=QD(3,5) QD(5,4)=QD(4,5)	CD(4,5)=P3*(G3/FACT3**2-G2/FACT13)-P5*G3/FACT3 QD(5,5)=P3*(G3/FACT3**2-G2/FACT13)-P5*G3/FACT3	
110	C	LOOP 2 IS FOR DUPLICATING THE SIMILAR PORTIONS OF AD DO 2 I=3,5 DO 2 J=3,5 QD(I+3,J+3)=QD(I,J)	CONTINUE SET QDPAR TO NONZERO PARTITION OF AD	

SUBROUTINE TBLTH 74/74 DBI=1 PWMPW FIN 4.8+528 89/21/81 12:15:51 PAGE 3

ETN 4.8+528 09/21/81 12:15:51

PMDMP 11/18/74 OPT 1

```

115      DO 43 I=1,6
        DO 43 J=1,6
          QDPAR(I,J)=QD(I+2,J+2)
C           TAKE SORT OF NONZERO PARTITION
C           CALL CHOLY(QDPAR,6,QDPRT)
        DO 44 I=1,8
          DO 44 J=1,8
            QDROOT(I,J)=0.
        DO 45 I=1,6
          DO 45 J=1,6
            QDROOT(I+2,J+2)=QDPRT(I,J)
            FORMAT(1X,8E9.2)
        DO 3  I=1,2
          DO 3  J=1,8
            H(I,J)=0.
            H(1,1)=1.
            H(1,3)=1.
            H(1,4)=1.
            H(2,2)=1.
            H(2,6)=1.
            H(2,7)=1.
130      RETURN
135

```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
TRUTH

RELOCATION		F.P.		F.P.		F.P.		TYPE		ARGS	
VARIABLES	SN	TYPE									
317 A		REAL									
0 DT		REAL									
324 FACT1		REAL									
325 FACT3		REAL									
340 G2		REAL									
0 H		REAL									
322 J		INTEGER									
316 NSTATE		INTEGER									
332 P1		REAL									
334 P3		REAL									
336 P5		REAL									
437 QDPAR		REAL									
0 QDROOT		REAL									
0 TD		REAL									
EXTERNS:											

STATEMENT LABELS

0 3
0 0 45

207

PAGE 4

09/21/81 12.15.51

FTN 4.8+528

SUBROUTINE TRUTH

PMDMP

DPT=1

FROM-TO

PROPERTIES

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
15	1	I	65 68	148	NOT INNER
23	1	J	66 68	38	INSTACK
177	2	I	110 113	129	NOT INNER
203	2	J	111 113	38	INSTACK
212	43	I	115 117	158	NOT INNER
220	43	J	116 117	38	INSTACK
231	44	I	120 122	128	NOT INNER
235	44	J	121 122	28	INSTACK
244	45	I	123 125	158	NOT INNER
252	45	J	124 125	38	INSTACK
262	3	I	127 129	128	NOT INNER
266	3	J	128 129	28	INSTACK

STATISTICS

PROGRAM LENGTH
52000B CM USED

555B

365

```

1      SUBROUTINE PROP(PHIT,QDROOT,H,XT,YT,N,M)
      REAL PHIT(N,N),QDROOT(N,N),XT(N,1),YT(M,1),H(M,N)
      REAL TEMP1(8,1),TEMP2(8,1)

```

```

5      C THIS ROUTINE IMPLEMENTS THE STATE TRANSITION EQUATION.
      C
      C XT(I+1)=PHIT*XT(I) + QDROOT*WD

```

```

10     C WHERE   XT= STATE VECTOR (NX1)
      C          PHIT= STATE TRANSITION MATRIX (NXN)
      C          QDROOT= STATE UNCERTAINTY COVARIANCE MATRIX (NXN)
      C          WD= GAUSSIAN DISTRIBUTED NOISE VECTOR (NX1)

```

```

15     C AND THE OUTPUT EQUATION
      C
      C YT=H*XT

```

```

20     C WHERE   YT=MEASUREABLE OUTPUT VECTOR (MX1)
      C          H= STATE TO OUTPUT MATRIX (MXN)

```

```

      C
      C CALL NOISE(TEMP1,N)
      C CALL MULT(QDROOT,TEMP1,N,N,1,TEMP2)
      C CALL MULT(PHIT,XI,N,N,1,TEMP1)
      C DO 1 I=1,N
      C 1 XT(I,1)=TEMP1(I,1)+TEMP2(I,1)
      C          ADD DETERMINISTIC VELOCITY OF 3 PIXELS OVER ANY TIME HISTORY IN
      C          THE X DIRECTION
      C XT(1,1)=XT(1,1)+.15
      C CALL MULT(H,XT,M,N,1,YT)
      C RETURN
      C END

```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 PROP

VARIABLES	SN	TYPE	RELOCATION	100	I	INTEGER
0 H		REAL	F.P.	0	N	INTEGER
0 N		INTEGER	F.P.	0	QDROOT	REAL
C PHIT		REAL	ARRAY	0	TEMP2	REAL
101 TEMP1		REAL	ARRAY	111	YT	REAL
0 XT		REAL	ARRAY	F.P.		ARRAY
EXTERNALS		TYPE	ARGS	6		F.P.
MULT			NOISE			
STATEMENT LABELS						
0 1						

SUBROUTINE IDEAL

74/74 OPT=1 PMODMP

FTN 4.8+528

PAGE 1

1 SUBROUTINE IDEAL(IMAX,S,XMAX,YMAX,N,NZ,X,Y,DATA,DX,DY)

REAL IMAX(3),XMAX(3),YMAX(3),S(12)

COMPLEX DATA(N,N),DX(N,N),DY(N,N)

DO 10 I=1,N

DO 10 J=1,N

DATA(I,J)=0.

DX(I,J)=0.

DY(I,J)=0.

CONTINUE

10 IF(N.LT.8) N=8

IF((N-8)/2.LT.NZ) NZ=(N-8)/2

LM=NZ+1

LP=N-NZ

LW=LW+1

DO 1 I=LW,LP

DO 1 J=LM,LW

X1=X+FLOAT(J-I*B)-XMAX(1)+.5

X2=X+FLOAT(J-I*B)-XMAX(2)+.5

X3=X+FLOAT(J-I*B)-XMAX(3)+.5

Y1=Y+FLOAT(I-I*B)-YMAX(1)+.5

Y2=Y+FLOAT(I-I*B)-YMAX(2)+.5

Y3=Y+FLOAT(I-I*B)-YMAX(3)+.5

ARG1=-.5*((X1*.2*S(1))+(X1+Y1)*(S(2)+S(3))+

(Y1**2*S(4)))

ARG2=-.5*((X2*.2*S(5))+(X2+Y2)*(S(6)+S(7))+

(Y2**2*S(8)))

ARG3=-.5*((X3*.2*S(9))+(X3+Y3)*(S(10)+S(11))+

(Y3**2*S(12)))

DATA(I,J)=IMAX(1)*EXP(ARG1)+IMAX(2)*EXP(ARG2)+IMAX(3)*EXP(ARG3)

ARG4=-X1*S(1)-.5*(S(2)+S(3))

ARG5=-X2*S(5)-.5*(S(6)+S(7))

ARG6=-X3*S(3)-.5*(S(10)+S(11))

ARG7=-Y1*S(4)-.5*(S(2)+S(3))

ARG8=-Y2*S(8)-.5*(S(6)+S(7))

ARG9=-Y3*S(12)-.5*(S(10)+S(11))

DX(I,J)=ARG4*IMAX(1)*EXP(ARG1)+ARG5*IMAX(2)*EXP(ARG2)

DX(I,J)=DX(I,J)+ARG6*IMAX(3)*EXP(ARG3)

DY(I,J)=ARG7*IMAX(1)*EXP(ARG1)+ARG8*IMAX(2)*EXP(ARG2)

DY(I,J)=DY(I,J)+ARG9*IMAX(3)*EXP(ARG3)

CONTINUE

RETURN

END

1

211

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 IDEAL

VARIABLES	SN	TYPE	RELOCATION	
2:6 ARG1		REAL	257 ARG2	REAL
2:0 ARG3		REAL	261 ARG4	REAL
2:2 ARG5		REAL	263 ARG6	REAL
2:4 ARG7		REAL	265 ARG8	REAL
2:6 ARG9		REAL	0 DATA	COMPLEX
0 DX		COMPLEX	0 DY	ARRAY
2:3 I		INTEGER	247 1B	INTEGER

2

09/21/81 12:15:51

FTN 4.8+52B

SUBROUTINE IDEAL

74/74 OFT=1 PWDMP

PAGE

VARIABLES	SN	TYPE	RELOCATION	
0 I _{MAX}		REAL	ARRAY F.P.	244 J
245 L ₁		INTEGER		246 LP
		INTEGER		0 N2
		INTEGER		0 X
0 N		REAL	ARRAY F.P.	250 X1
0 S		REAL	ARRAY F.P.	252 X3
0 X _{MAX}		REAL		0 YMAX
251 X ₂		REAL		254 Y2
		REAL		
0 Y		REAL		
253 Y ₁		REAL		
255 Y ₃		REAL		

EXTERNALS	TYPE	ARGS
EXP	REAL	1 LIBRARY

INLINE FUNCTIONS	TYPE	ARGS
FLOAT	REAL	1 INTRIN

STATEMENT LABELS

LOCPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
11	10	1	4 9	16B	NOT INNER
20	10	J	5 9	4B	INSTACK
45	1	I	15 37	172B	EXT REFS NOT INNER
47	1	J	16 37	165B	EXT REFS

21 STATISTICS
PROGRAM LENGTH 52000B CM USED
207

21

SUBROUTINE SMOOTH 74/74 OPT=1 PMOMP FTN 4.8+528 09/21/81 12.15.51 PAGE 1

```
1            SUBROUTINE SMOOTH(DATA,SDATA,ALPHA,N,ITERA)
2            COMPLEX DATA(N,N),SDATA(:,N)
3            C THIS ROUTINE SMOOTH'S RAW DATA ARRAY DATA USING EXPONENTIAL
4            C SMOOTHING. WEIGHTING FACTOR ALPHA IS USED TO GENERATE THE
5            C SMOOTHED DATA IN ARRAY SDATA. THE PARAMETER ITERA IS
6            C USED TO DETERMINE THE WEIGHTING FACTOR WHEN FEWER THAN
7            C 1/ALPHA ITERATIONS HAVE BEEN DONE.
8            A=1./ITERA
9            IF(A.LT.ALPHA) A=ALPHA
10          DO 3 I=1,N
11          DO 3 J=1,N
12          SDATA(I,J)=A*DATA(I,J)+(1.-A)*SDATA(I,J)
13          RETURN
14          END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

213	VARIABLES	SN	TYPE	RELOCATION	0	ALPHA	REAL	F. P.
	45 A		REAL		46 1	1	INTEGER	
	0 DATA		COMPLEX	F.P.	47 J	1	INTEGER	
	C ITERA		INTEGER	F.P.	0 SDATA	0	COMPLEX	
	0 N		INTEGER	F.P.			ARRAY	F. P.
STATEMENT LABELS	0 1		INACTIVE	0 3				
LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES			
16	3	I	10 12	23B	NOT INNER			
30	3	J	11 12	6B	INSTACK			
STATISTICS								
PROGRAM LENGTH	52000B	CM USED	56B	46				

SUBROUTINE NOISE 74/74 OPT=1 PMDMP

FTN 4.8+528 09/21/81 12.15.51

PAGE 1

```
1      SUBROUTINE NOISE(W,N)
      REAL W(N)
      C THESE STATEMENTS WERE USED IN THE MODCOMP VERSION OF THE SOFTWARE
      C DATA FIRST/0/
      C IF(IFIRST.NE.0) GO TO 1
      C IA=12345
      C IFIRST=1
      IA=1
      DO 2 I=1,N
      CALL GAUSS(IA,IY,VAL)
      W(I)=VAL
      IA=IY
      CONTINUE
      RETURN
      END
2
10
15
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3 NOISE

VARIABLES	SN	TYPE	RELOCATION	24	IA	INTEGER
25 I		INTEGER		0	N	INTEGER
26 IY		INTEGER		0	W	REAL
27 VAL		REAL				ARRAY

EXTERNALS TYPE ARGS

GAUSS

STATEMENT LABELS

0 1 INACTIVE

LCOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES	EXT REFS
10 2	1	9 13	76			

STATISTICS

PROGRAM LENGTH 33B 27
 52000B CM USED


```

C      WRITE(6,101) X1,Y1,S(1),S(2),S(3),S(4),IMAX(1),ARG1
101    FORMAT(2X,6E12.5)
       ARG2=-.5*(X2**2*S(5))+(X2*Y2)*(S(6)+S(7))+(Y2**2*S(8)))
       ARG3=-.5*((X3**2*S(9))+(X3*Y3)*(S(10)+S(11))+(Y3**2*S(12)))
       FXY=IMAX(1)*EXP(ARG1)+IMAX(2)*EXP(ARG2)+IMAX(3)*EXP(ARG3)
       AVG=AVG+FXY
       SUMXY=SUMX+XP*FXY
       SUMY=SUMY+YP*FXY
       CONTINUE
3      DATA(I,J)=AVG/25.
       WRITE(6,100) I,J,DATA(I,J)
100    FORMAT(2X,2I4,2X,E12.5)
       SUMAVG=SUMAVG+AVG
       CONTINUE
2      CENX=SUMX/SUMAVG
       CENY=SUMY/SUMAVG
       RETURN
1      END
75
65      4      CONTINUE
3      DATA(I,J)=AVG/25.
       WRITE(6,100) I,J,DATA(I,J)
100    FORMAT(2X,2I4,2X,E12.5)
       SUMAVG=SUMAVG+AVG
       CONTINUE
2      CENX=SUMX/SUMAVG
       CENY=SUMY/SUMAVG
       RETURN
1      END

```

SYMBOLIC REFERENCE MAP (R=1)

216 ENTRY POINTS
3 INPUT3

VARIABLES	SN	TYPE	RELOCATION						
221 ARG1		REAL		222 ARG2					
223 ARG3		REAL		204 AVG					
0 CENX		REAL	F.P.	0 CENY					F.P.
0 DATA		COMPLEX	ARRAY	210 DELX					
207 DELY		REAL	F.P.	224 FXY					
175 1		INTERFACE	ARRAY	0 IMAX					ARRAY
176 J		INTEGER	K1	205 K1					INTEGER
206 K2		INTEGER	L1	202 L1					
203 LP		INTEGER	O	0 N					INTEGER
0 S		REAL	ARRAY	201 SUMAVG					
177 SUMX		REAL	F.P.	200 SUMY					
0 X		REAL	ARRAY	0 XMAX					REAL
211 XP		REAL	F.P.	213 X1					
215 X2		REAL		217 X3					REAL
0 Y		REAL	F.P.	0 YMAX					REAL
212 YP		REAL		214 Y1					
216 Y2		REAL		220 Y3					REAL
EXTERNALS		TYPE	ARGS						
EXP		REAL	1 LIBRARY						
STATEMENT LABELS									
0 1			0 2						
0 4			0 10						
1E.4 101			FMT NO REFS						
				166	0	3			
					100		FMT	NO REFS	

PAGE 3

SUBROUTINE INPUT3 74/74 OPT=1 PMDMP

FTN 4.8+528

09/21/81 12.15.51

LOCPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
11	10	1	33 35	13B	NOT INNER
17	10	J	34 35	2B	INSTACK
32	1	I	41 73	122B	EXT REFS
34	2	J	42 72	116B	EXT REFS
36	3	K1	45 67	103B	NOT INNER
37	4	K2	46 66	77B	EXT REFS

STATISTICS
PROGRAM LENGTH 240B 160
52000B CM USED

SUBROUTINE DISPLAY	74/74	OPT=1	PMDMP	FTN 4.8+528	09/21/81	12.15.51	PAGE
1							1
	SUBROUTINE DISPLAY(IXSIZE,IYSIZE,DATA)						
	INTEGER IXY(122)						
	COMPLEX DATA(IYSIZE,IXSIZE)						
5	WRITE(6,102)						
102	FORMAT('1H')						
	NUM=IXSIZE+2						
	DO 5 I=1,NUM						
5	IXY(I)=1H-						
	WRITE(6,103) (IXY(I),I=1,NUM)						
103	FORMAT('1H',122A1)						
	DMIN=1.E30						
	DMAX=-1.E30						
10	DO 1 I=1,IXSIZE						
	DO 1 J=1,IYSIZE						
	DMAX=AMAX1(DMAX,CABS(DATA(J,I)))						
	DMIN=AMIN1(DMIN,CABS(DATA(J,I)))						
15	DO 4 J=1,IYSIZE						
	DO 2 I=1,IXSIZE						
	IXY(I)=1H						
	X=(CABS(DATA(J,I))-DMIN)/(DMAX-DMIN)						
	IF(X.GT.-.42) IXY(I)=1H0						
	IF((X.GT..28).AND.(X.LE.-.42)) IXY(I)=1HX						
	IF((X.GT.-.14).AND.(X.LE.-.28)) IXY(I)=1H+						
2	CONTINUE						
	NUM=IXSIZE+1						
	IXY(NUM)=1H						
	WRITE(6,100) (IXY(I),I=1,NUM)						
100	FORMAT('1A.*',121A1)						
	DO 3 I=1,IXSIZE						
	IXY(I)=1H						
	X=(CABS(DATA(J,I))-DMIN)/(DMAX-DMIN)						
	IF((X.GT..56).AND.(X.LE..70)) IXY(I)=1H-						
	IF((X.GT..70).AND.(X.LE..84)) IXY(I)=1H+						
	IF((X.GT..84)) IXY(I)=1H#						
3	CONTINUE						
	WRITE(6,101) (IXY(I),I=1,IXSIZE)						
101	FORMAT('1H+,15,120A1)						
4	CONTINUE						
	NUM=IXSIZE+2						
	DO 6 I=1,NUM						
6	IXY(I)=1H-						
	WRITE(6,103) (IXY(I),I=1,NUM)						
45	RETURN						
	END						

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 DISPLAY

SUBROUTINE DISPLAY

74/74 OPT=1 PMDMP

VARIABLES	SN	TYPE	COMPLEX	RELLOCATION
0 DATA		REAL	ARRAY F.P.	262 DMAX
261 DMIN		REAL	ARRAY F.P.	260 I
0 IXSIZE		INTEGER	ARRAY F.P.	265 IXY
0 IYSIZE		INTEGER	ARRAY F.P.	263 J
257 NUM		INTEGER	REAL	264 X

FILE NAMES	MODE	FMT
TAPE6		

EXTERNALS	TYPE	ARGS
CABS	REAL	1 LIBRARY

INLINE FUNCTIONS	TYPE	ARGS
AMAX1	REAL	0 INTRIN

STATEMENT LABELS			
------------------	--	--	--

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES	INSTACK	EXT REFS	NOT INNER
15	5	1	7 8	33				
32	1	1	13 16	243				
33	1	J	14 16	21B				
57	4	J	17 38	110B				
60	2	I	18 24	345				
124	3	I	29 35	33B				
173	6	I	40 41	3B	INSTACK			

STATISTICS	PROGRAM LENGTH	473B	315
	520COB CM USED		

SUBROUTINE	SHIFT	74/74	OPT=1	PMDMP	FTN 4.8+528	09/21/81	12.15.51	PAGE
1								1
1	SUBROUTINE SHIFT(DATA,N,XSHIFT,YSHIFT)							
	COMPLEX DATA(N,N),TEMP1,TEMP2,TEMP3,TEMP4,FX,FYC							
5	C THIS ROUTINE IMPLEMENTS A SPATIAL PHASE SHIFT							
	C IN THE FREQUENCY DOMAIN. THE ARRAY DATA IS ASSUMED TO BE THE							
	C NXN ARRAY OF FOURIER TRANSFORM COMPONENTS AS GENERATED BY THE							
	C TRANSFORM ROUTINE FOURT. FOR EXAMPLE, FOR A 6X6 ARRAY DATA							
10	C							
	C	X0 Y0 X1 Y0 X2 Y0 X3 Y0 X2* Y0 X1* Y0						
	C	X0 Y1 X1 Y1 X2 Y1 X3 Y1 X2* Y1 X1* Y1						
15	C	X0 Y2 X1 Y2 X3 Y2 X3 Y2 X2* Y2 X1* Y2						
	C	X0 Y3 X1 Y3 X2 Y3 X3 Y3 X2* Y3 X1* Y3						
	C	X0 Y2* X1 Y2* X2 Y2* X3 Y2* X2* Y2* X1* Y2*						
20	C	X0 Y1* X1 Y1* X2 Y1* X3 Y1* X2* Y1* X1* Y1*						
25	C							
	C	PHASE SHIFTING IS IMPLEMENTED BY MULTIPLYING THE						
	C	FOURIER TRANSFORM COMPONENTS BY						
	C	EXP(J*2*PI*(FX*XSHIFT+FY*YSHIFT))						
30	C							
	C	XSHIFT AND YSHIFT ARE THE SHIFTS IN THE X AND Y COORDINATE						
	C	DIRECTIONS.						
35	C							
	C	P1=3.141592654						
	C	DEM=FLOAT(N)						
	C	NCENT=N/2+1						
	C	DO 1 I=1,NCENT						
	C	DO 1 J=1 NCENT						
	C	FX=CMPLX(0.,-2.*P1*(J-1)*XSHIFT/DEM)						
	C	FY=CMPLX(0.,-2.*P1*(I-1)*YSHIFT/DEM)						
	C	FXC=CONJG(FX)						
	C	FYC=CONJG(FY)						
	C	TEMP1=DATA(I,J)						
	C	DATA(I,J)=TEMP1*CEXP(FX+FY)						
40	C	IF(I.EQ.1) GO TO 10						
	C	IF(J.EQ.1.OR.J.EQ.NCENT) GO TO 20						
	C	TEMP2=DATA(I,N+2-J)						
	C	TEMP3=DATA(N+2-I,J)						
	C	TEMP4=DATA(N+2-I,N+2-J)						
45	C	DATA(I,N+2-J)=TEMP2*CEXP(FXC+FYC)						
	C	DATA(N+2-I,J)=TEMP3*CEXP(FX+FYC)						
	C	DATA(N+2-I,N+2-J)=TEMP4*CEXP(FXC+FYC)						
	C	GO TO 1						
	C	IF(J.EQ.1.OR.J.EQ.NCENT) GO TO 1						
50	C	TEMP2=DATA(I,N+2-J)						
	C	TEMP3=DATA(N+2-I,J)						
	C	DATA(I,N+2-J)=TEMP2*CEXP(FXC+FYC)						
	C	GO TO 1						
55	C	TEMP3=DATA(N+2-I,J)						
	C	DATA(N+2-J)=TEMP3*CEXP(FXC+FYC)						
	C	CONTINUE						
	C	RETURN						

SUBROUTINE SHIFT

74/74 OPT=1 PWDMP

END

FTN 4.8+528

09/21/81

12.15.51

PAGE 2

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINT'S
3 SHIFT

VARIABLES	SN	TYPE	RELOCATION	
0 DATA		COMPLEX	ARRAY F.P.	
231 FX		COMPLEX		REAL
235 FY		COMPLEX		COMPLEX
2:4 I		INTEGER		COMPLEX
0 N		INTEGER		INTEGER
241 PI		REAL		INTEGER
223 TEMP2		COMPLEX		INTEGER
227 TEMP4		COMPLEX		TEMP1
0 YSHIFT		REAL		TEMP3
EXTERNALS		TYPE	ARGS	
CEXP		COMPLEX	1 LIBRARY	
INLINE FUNCTIONS		TYPE	ARGS	
CMPX		COMPLEX	2 INTRIN	CONJG
FLOAT		REAL	1 INTRIN	COMPLEX
STATEMENT LABELS				INTRIN
210 1			0 7	INACTIVE
166 20				137 10
LOOPS	LABEL	INDEX	FROM-TO	PROPERTIES
16 1	I	I	33 56	LENGTH
17 1	J	J	34 56	EXT REFS
STATISTICS			177B	NOT INNER
PROGRAM LENGTH			174B	EXT REFS
52000B CM USED			275B	
			189	

SUBROUTINE DERIV 74/74 OPT=1 PMODMP

FTN 4.8+529

09/21/81 12.15.51 PAGE 1

1 SUBROUTINE DERIV(DATA,N,DX,DY)
2 COMPLEX DATA(N,N),TEMP1,TEMP2,TEMP3,TEMP4,FX,FYC
3 COMPLEX DX(N,N),DY(N,N)

4 THIS ROUTINE IMPLEMENTS SPATIAL PARTIAL DERIVATIVES
5 IN THE FREQUENCY DOMAIN. THE ARRAY DATA IS ASSUMED TO BE THE
6 NXN ARRAY OF FOURIER TRANSFORM COMPONENTS AS GENERATED BY THE
7 TRANSFORM ROUTINE FOURT. FOR EXAMPLE, FOR A 6X6 ARRAY DATA

8
9 | X0 Y0 | X1 Y0 | X2 Y0 | X3 Y0 | X2* Y0 | X1* Y0 |
10 | X0 Y1 | X1 Y1 | X2 Y1 | X3 Y1 | X2* Y1 | X1* Y1 |
11 | X0 Y2 | X1 Y2 | X3 Y2 | X3 Y2 | X2* Y2 | X1* Y2 |
12 | X0 Y3 | X1 Y3 | X2 Y3 | X3 Y3 | X2* Y3 | X1* Y3 |
13 | X0 Y2* | X1 Y2* | X2 Y2* | X3 Y2* | X2* Y2* | X1* Y2* |
14 | X0 Y1* | X1 Y1* | X2 Y1* | X3 Y1* | X2* Y1* | X1* Y1* |

15 DIFFERENTIATION IS IMPLEMENTED BY MULTIPLYING THE
16 FOURIER TRANSFORM COMPONENTS BY
17 J*2*PI*FX AND J*2*PI*FY

20 PI=3.141592654
21 DEM=FLOAT(N)
22 NCENT=N/2+1

23 DO 1 I=1,NCENT
24 DO 1 J=1,NCENT
25 FX=CMPXLX(0.,+2.*PI*(J-1)/DEM)
26 FY=CMPXLX(0.,+2.*PI*(I-1)/DEM)
27 FXC=CONJG(FX)
28 FYC=CONJG(FY)

29 TEMP1=DATA(1,J)
30 DX(1,J)=TEMP1*FX
31 Dr(1,J)=TEMP1*FY
32 IF(1.EQ.1) GO TO 10
33 IF(J.EQ.1.OR.J.EQ.NCENT) GO TO 20

34 TEMP2=DATA(1,N+2-J)
35 TEMP3=DATA(N+2-1,J)
36 TEMP4=DATA(N+2-1,N+2-J)
37 DX(1,N+2-J)=TEMP2*FX
38 DY(1,N+2-J)=TEMP3*FY
39 DX(N+2-1,J)=TEMP3*FX
40 DY(N+2-1,J)=TEMP3*FYC
41 DX(N+2-1,N+2-J)=TEMP4*FYC
42 GO TO 1

43 IF(J.EQ.1.OR.J.EQ.NCENT) GO TO 1
44 TEMP2=DATA(1,N+2-J)
45 DX(1,N+2-J)=TEMP2*FY
46 CY(1,N+2-J)=TEMP2*FC
47 DY(1,N+2-J)=TEMP2*FY
48 GO TO 1

49 TEMP3=DATA(N+2-1,J)
50 TEMP4=DATA(N+2-1,N+2-J)
51 DX(1,N+2-J)=TEMP4*FYC
52 GO TO 1

53 IF(J.EQ.1.OR.J.EQ.NCENT) GO TO 1
54 TEMP2=DATA(1,N+2-J)
55 CY(1,N+2-J)=TEMP2*FC
56 DY(1,N+2-J)=TEMP2*FY
57 GO TO 1

58 TEMP3=DATA(N+2-1,J)
59 TEMP4=DATA(N+2-1,N+2-J)
60 DX(1,N+2-J)=TEMP4*FYC
61 GO TO 1

SUBROUTINE DERIV 74/74 OPT=1 PWDMP

FTN 4.8+528 09/21/81 12.15.51 PAGE

```
DX(N+2-I,J)=TEMP3*FX
DY(N+2-I,J)=TEMP3*FYC
CONTINUE
RETURN
END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 DERIV

VARIABLES	SN	TYPE	RELOCATION	F.P.	DEM	REAL	ARRAY	F.P.
0 DATA		COMPLEX	ARRAY	F.P.	225	DEM	COMPLEX	
0 DX		COMPLEX	ARRAY	F.P.	0	DY	COMPLEX	
214 FX		COMPLEX			216	FXC	COMPLEX	
220 FY		COMPLEX			222	FYC	COMPLEX	
227 I		INTEGER			230	J	INTEGER	
0 N		INTEGER			226	NCNT	INTEGER	
224 P1		REAL			204	TEMP1	COMPLEX	
206 TEMP2		COMPLEX			210	TEMP3	COMPLEX	
212 TEMP4		COMPLEX						
INLINE FUNCTIONS		TYPE	ARGS		CONJG	COMPLEX	1 INTRIN	
		COMPLEX	2 INTRIN					
		REAL	1 INTRIN					
STATEMENT LABELS					0	7	INACTIVE	124 10
162 L	1							
146 L	20							
LOOPS	LABEL	INDEX	FROM-TO	LENGTH		PROPERTIES		
16	1	I	31 60	151B		NOT INNER		
36	1	J	32 60	126B	OPT			
STATISTICS	PROGRAM LENGTH	260B	176					
	52000B CM USED							

SUBROUTINE FOURT(DATA,NN,NDIM,ISIGN,IFORM,WORK)
FOR INFORMATION CONTACT MR. MARK HALLER 4950/ADDS/56248

THE COOLEY-TUKEY FAST FOURIER TRANSFORM IN USASI BASIC FORTRAN

5

```

C   TRANSFORM(K1,K2,...) = SUM(DATA(J1,J2,...)*EXP((ISIGN*2*PI*SQRT(-1))*FFT0030
C   * ((J1-1)*(K1-1)/NN(1)+(J2-1)*(K2-1)/NN(2)+...)). SUMMED FOR ALL FFT0040
C   J1, K1 BETWEEN 1 AND NN(1), J2, K2 BETWEEN 1 AND NN(2), ETC. FFT0050
C   THERE IS NO LIMIT TO THE NUMBER OF SUBSCRIPTS. DATA IS A FFT0060
C   MULTIDIMENSIONAL COMPLEX ARRAY WHOSE REAL AND IMAGINARY FFT0070
C   PARTS ARE ADJACENT IN STORAGE, SUCH AS FORTRAN IV PLACES THEM. FFT0080
C   IF ALL IMAGINARY PARTS ARE ZERO (DATA ARE DISGUISED REAL), SET FFT0090
C   IFORM TO ZERO TO CUT THE RUNNING TIME BY UP TO FORTY PERCENT. FFT0100
C   OTHERWISE, IFORM = +1. THE LENGTHS OF ALL DIMENSIONS ARE FFT0110
C   STORED IN ARRAY NIN, OF LENGTH NDIM. THEY MAY BE ANY POSITIVE FFT0120
C   INTEGERS. TWO THE PROGRAM RUNS FASTER ON COMPOSITE INTEGERS, AND FFT0130
C   ESPECIALLY FAST ON NUMBERS RICH IN FACTORS OF TWO. ISIGN IS +1 FFT0140
C   OR -1. IF A -1 TRANSFORM IS FOLLOWED BY A +1 ONE (OR A +1 FFT0150
C   BY A -1) THE ORIGINAL DATA REAPPEAR, MULTIPLIED BY NNOT (=NN(1)* FFT0160
C   NN(2)*...). TRANSFORM VALUES ARE ALWAYS COMPLEX, AND ARE RETURNED FFT0170
C   IN ARRAY DATA, REPLACING THE INPUT. IN ADDITION, IF ALL FFT0180
C   DIMENSIONS ARE NOT POWERS OF TWO, ARRAY WORK MUST BE SUPPLIED. FFT0190
C   COMPLEX OF LENGTH EQUAL TO THE LARGEST NIN 2**K DIMENSION. FFT0200
C   OTHERWISE, REPLACE WORK BY ZERO IN THE CALLING SEQUENCE. FFT0210
C   NORMAL FORTRAN DATA ORDERING IS EXPECTED, FIRST SUBSCRIPT VARYING FFT0220
C   FASTEST. ALL SUBSCRIPTS BEGIN AT ONE. FFT0230
C   FFT0240
C   FFT0250
C   FFT0260
C   FFT0270
C   FFT0280
C   FFT0290
C   FFT0300
C   FFT0310
C   FFT0320
C   FFT0330
C   FFT0340
C   FFT0350
C   FFT0360
C   FFT0370
C   FFT0380
C   FFT0390
C   FFT0400
C   FFT0410
C   FFT0420
C   FFT0430
C   FFT0440
C   FFT0450
C   FFT0460
C   FFT0470
C   FFT0480
C   FFT0490
C   FFT0500
C   FFT0510
C   FFT0520
C   FFT0530
C   FFT0540

```

RUNNING TIME IS MUCH SHORTER THAN THE NAIVE NNOT**2, BEING
GIVEN BY THE FOLLOWING FORMULA. DECOMPOSE NNOT INTO
 $2^*K2 * 3**K3 * 5**K5 * ...$ LET SUM12 = $2*K2$, SUMF = $3*K3 + 5*K5$
+ ... AND NF = $K3 + K5 + ...$. THE TIME TAKEN BY A MULTIDI-
MENSIONAL TRANSFORM ON THESE NNOT DATA IS $T = T0 + NNOT * (T1 +$
 $T2 * SUM2 + T3 * SUM4 + T4 * NF)$. ON THE CDC 3300 (FLOATING POINT ADD TIME
OF SIX MICROSECONDS), $T = 3000 + NNOT * (500 + 43 * SUM2 + 68 * SUMF +$
 $320 * NF)$ MICROSECONDS ON COMPLEX DATA. IN ADDITION, THE
ACCURACY IS GREATLY IMPROVED. AS THE RMS RELATIVE ERROR IS
BOUNDED BY $3^{*}2^{*}(-B)$ SUM(FACTOR(j)**1.5), WHERE B IS THE NUMBER
OF BITS IN THE FLOATING POINT FRACTION AND FACTOR(j) ARE THE
PRIME FACTORS OF NNOT.

PROGRAM BY NORMAN BRENNER FROM THE BASIC PROGRAM BY CHARLES
RADER. RALPH ALTER SUGGESTED THE IDEA FOR THE DIGIT REVERSAL.
MIT LINCOLN LABORATORY, AUGUST 1967. THIS IS THE FASTEST AND MOST EFFICIENT
VERSATILE VERSION OF THE FFT KNOWN TO THE AUTHOR. SHORTER PROGRAMS
FOUR1 AND FOUR2 RESTRICT DIMENSION LENGTHS TO POWERS OF TWO. FFT0430
SEE - IEEE AUDIO TRANSACTIONS (JUNE 1967), SPECIAL ISSUE ON FFT. FFT0440
THE DISCRETE FOURIER TRANSFORM PLACES THREE RESTRICTIONS UPON THE FFT0450
DATA.
1. THE NUMBER OF INPUT DATA AND THE NUMBER OF TRANSFORM VALUES FFT0470
MUST BE THE SAME.
2. BOTH THE INPUT DATA AND THE TRANSFORM VALUES MUST REPRESENT FFT0490
EQUISPACED POINTS IN THEIR RESPECTIVE DOMAINS OF TIME AND FFT0500
FREQUENCY. CALLING THESE SPACINGS DELTAT AND DELTAF, IT MUST BE FFT0510
TRUE THAT $\text{DELTAF} = 2 \cdot \text{PI} / (\text{NIN}(1) * \text{DELTAT})$. OF COURSE, DELTAT NEED NOT FFT0520
BE THE SAME FOR EVERY DIMENSION.

40

45

50

74/74 OPT=1 PMDMP

FTN 4.8+528

PAGE 2

C 3. CONCEPTUALLY AT LEAST, THE INPUT DATA AND THE TRANSFORM OUTPUT FFT T0550
C REPRESENT SINGLE CYCLES OF PERIODIC FUNCTIONS.
C
60 C EXAMPLE 1. THREE-DIMENSIONAL FORWARD FOURIER TRANSFORM OF A
C COMPLEX ARRAY DIMENSIONED 32 BY 25 BY 13 IN FORTRAN IV.
C DIMENSION DATA(32,25,13),WORK(50),NN(3)
C COMPLEX DATA
C DATA NN/32,25,13/
C DO 1 I=1,32
C DO 1 J=1,25
C DO 1 K=1,13
C 1 DATA(I,J,K)=COMPLEX VALUE
C CALL FOURT(DATA,NN,3,-1,1,WORK)
C
C EXAMPLE 2. ONE-DIMENSIONAL FORWARD TRANSFORM OF A REAL ARRAY OF
C LENGTH 64 IN FORTRAN II.
C DIMENSION DATA(2,64)
C DO 2 I=1,64
C 2 DATA(1,I)=REAL PART
C DATA(2,I)=0.
C CALL FOURT(DATA,64,1,-1,0,0)
C
C DIMENSION DATA(1),NN(1),IFACT(32),WORK(1)
C
80 C CRC 6600 INITIALIZATION
C WR=0.
C WI=0.
C WSTPR=0.
C WSPI=0.
C
85 T2OPI=6.283185307
C IF(NDIM-1)920,1,1
C NTOT=2
1 DO 2 IDIM=1,NDIM
C IF(NN(IDIM))920,920,2
C 2 NTOT=NTOT*NN(IDIM)
C
C MAIN LOOP FOR EACH DIMENSION
C
95 N1=2
C D3 910 IDIM=1,NDIM
C N=N*(IDIM)
C N1=N1+N
C IF((N-1)*920,900,5
C
100 C C FACTOR N
C
5 M=N
C NTWO=NP1
C
105 LF=1
C IDIV=2
C ICUDI=N/IDIV
C ICUM=M-IDIV*ICUDI
C IF((ICUDI-IDIV)*ICUDI
C 11 IF((IREM)20,12,20
C 12 NWQ=NTWO+NTWO
C IQUDI=ICUDI
C GO TO 10
C IDIV=3

SUBROUTINE	FOURT	74/74	OPT=1	PMOMP	FTN 4.8+528	09/21/81	12.15.51	PAGE	3
115	30	IQUOT=M/IDIV IREM=M-IDIV*IQUOT IF((IQUOT-IDIV)>0,31,31 IF((IREM)>0,32,40 IFACT(1F)=IDIV IF=IF+1 N=IQUOT GO TO 30 IDIV=IDIV+2				FFT1070 FFT1080 FFT1090 FFT1100 FFT1110 FFT1120 FFT1130 FFT1140 FFT1150 FFT1160 FFT1170 FFT1180 FFT1190 FFT1200 FFT1210 FFT1220			
120	31				FFT11240				
	32				FFT11260				
	33				FFT11270				
	34				FFT11290				
	35				FFT1130				
	36				FFT1135				
	37				FFT11360				
	38				FFT11370				
	39				FFT11380				
	40				FFT11390				
	41				FFT11400				
	42				FFT11410				
	43				FFT11420				
	44				FFT11430				
	45				FFT11440				
	46				FFT11450				
	47				FFT11460				
	48				FFT11470				
	49				FFT11480				
	50				FFT11490				
	51				FFT11500				
	52				FFT11510				
	53				FFT11520				
	54				FFT11530				
	55				FFT11540				
	56				FFT11550				
	57				FFT11560				
	58				FFT11570				
	59				FFT11580				
	60				FFT11590				
	61				FFT11600				
	62				FFT11610				
	63				FFT11620				
	64				FFT11630				

SEPARATE FOUR CASES--

1. COMPLEX TRANSFORM OR REAL TRANSFORM FOR THE 4TH, 5TH, ETC. DIMENSIONS.
2. REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION. METHOD-- TRANSFORM HALF THE DATA, SUPPLYING THE OTHER HALF BY CONJUGATE SYMMETRY.
3. REAL TRANSFORM FOR THE 1ST DIMENSION, N ODD. METHOD-- TRANSFORM HALF THE DATA AT EACH STAGE, SUPPLYING THE OTHER HALF BY CONJUGATE SYMMETRY.
4. REAL TRANSFORM FOR THE 1ST DIMENSION, N EVEN. METHOD-- TRANSFORM A COMPLEX ARRAY OF LENGTH N/2 WHERE REAL PARTS ARE THE EVEN NUMBERED REAL VALUES AND WHOSE IMAGINARY PARTS ARE THE ODD NUMBERED REAL VALUES. SEPARATE AND SUPPLY THE SECOND HALF BY CONJUGATE SYMMETRY.

C NON2=NP1*(NP2/NTWO)
ICASE=1
IF((IDIM-4)>1,90,90
IF((IFORM)>1)72,72,90
71
72 ICASE=2
IF((IDIM-1)>3,73,90
ICASE=3
IF((NTWO-NP1)>0,90,74
73
74 ICASE=4
NTWO=NTWO/2
N=N/2
NP2=NP2/2
NTOT=NTOT/2
I=3
DO 80 J=2,NTOT
DATA(J)=DATA(1)
I=I+2
1=1+2
I1RNG=NP1
IF((ICASE-2)>100,95,100
95
I1RNG=NP0*(1+NPREV/2)

C SHUFFLE ON THE FACTORS OF TWO IN N. AS THE SHUFFLING CAN BE DONE BY SIMPLE INTERCHANGE, NO WORKING ARRAY IS NEEDED
C 100 IF((NTWO-NP1)>600,600,113
110 NP2HF=NP2/2

```

DO 150 I2=1,NP2,NON2
  IF((J-I2)>120,130,130
  I1MAX=I2+NON2-2
  DO 125 I1=I2,I1MAX,2
    DO 125 I3=I1,NTOT,NP2
      J3=J+3-I2
      TEMP=DATA(I3)
      TEMP1=DATA(I3+1)
      DATA(I3)=DATA(J3)
      DATA(I3+1)=DATA(J3+1)
      DATA(J3)=TEMP
      DATA(J3+1)=TEMP1
      M=IP2HF
      N=I-M
      IF(I-M)<150,150,145
      J=I-M
      M=M/2
      IF(I-M-NON2)<150,140,140
      J=I+M
      150 C
      MAIN LOOP FOR FACTORS OF TWO.  PERFORM FOURIER TRANSFORMS OF
      LENGTH FOUR, WITH ONE OF LENGTH TWO IF NEEDED.  THE TWIDDLE FACTOR
      W=EXP(1SIGN*2*PI*SQRT((-1)*N/(4*MMAX))).  CHECK FOR W=ISIGN*SQRT((-1))FTT1850
      AND REPEAT FOR W=ISIGN*SQRT((-1))*CONJUGATE(W).
      C
      NON2T=NON2+NON2
      IPAR=NTWO/NP1
      IF(IPAR-2)<350,330,320
      IPAR=IPAR/4
      GO TO 310
      310 DO 340 I1=1,11RNG,2
      DO 340 J3=I1,NON2,NP1
      DO 340 K1=J3,NTOT,NON2T
        K2=K1+NON2T
        1E:PR=DATA(K2)
        TEMP1=DATA(K2+1)
        DATA(K2)=DATA(K1)-TEMP1
        DATA(K2+1)=DATA(K1+1)-TEMP1
        DATA(K1)=DATA(K1)+TEMP1
        DATA(K1+1)=DATA(K1+1)+TEMP1
        K*MAX=40*N2
        350 IF((K*MAX-1)*P2HF>1370,600,600
        LMAX=M*NON2T,MMAX/2)
        IF(M*MAX-NON2T)<405,405,380
        THETA=TWOPI*FLOAT(NON2)/FLOAT(4*MMAX)
        IF(ISIGN)<400,390,390
        THETA=-THETA
        390 WR=COS(THETA)
        400 WI=SIN(THETA)
        410 WI=FR/-2.*WI*WI
        410 WR=WR*WR-WI*WI
        410 W2I=2.*WR*WI
        410 W3I=W2R*W6-W2I*WI
        410 W3I=W2R*WI+W2I*WR
        410 M=L
        410 IF((M*MAX-NON2)>420,420,410
        410 FTT1210
        410 FTT12130
        410 FTT12140
        410 FTT12150
        410 FTT12160
        410 FTT12170
        410 FTT12180
        410 FTT12190
        410 FTT12200
      220
      225
      230
      235
      240
      245
      250
      255
      260
      265
      270
      275
      280
      285
      290
      295
      300
      305
      310
      315
      320
      325
      330
      335
      340
      345
      350
      355
      360
      365
      370
      375
      380
      385
      390
      395
      400
      405
      410
      415
      420
      425
      430
      435
      440
      445
      450
      455
      460
      465
      470
      475
      480
      485
      490
      495
      500
      505
      510
      515
      520
      525
      530
      535
      540
      545
      550
      555
      560
      565
      570
      575
      580
      585
      590
      595
      600
      605
      610
      615
      620
      625
      630
      635
      640
      645
      650
      655
      660
      665
      670
      675
      680
      685
      690
      695
      700
      705
      710
      715
      720
      725
      730
      735
      740
      745
      750
      755
      760
      765
      770
      775
      780
      785
      790
      795
      800
      805
      810
      815
      820
      825
      830
      835
      840
      845
      850
      855
      860
      865
      870
      875
      880
      885
      890
      895
      900
      905
      910
      915
      920
      925
      930
      935
      940
      945
      950
      955
      960
      965
      970
      975
      980
      985
      990
      995
      1000
      1005
      1010
      1015
      1020
      1025
      1030
      1035
      1040
      1045
      1050
      1055
      1060
      1065
      1070
      1075
      1080
      1085
      1090
      1095
      1100
      1105
      1110
      1115
      1120
      1125
      1130
      1135
      1140
      1145
      1150
      1155
      1160
      1165
      1170
      1175
      1180
      1185
      1190
      1195
      1200
      1205
      1210
      1215
      1220
      1225
      1230
      1235
      1240
      1245
      1250
      1255
      1260
      1265
      1270
      1275
      1280
      1285
      1290
      1295
      1300
      1305
      1310
      1315
      1320
      1325
      1330
      1335
      1340
      1345
      1350
      1355
      1360
      1365
      1370
      1375
      1380
      1385
      1390
      1395
      1400
      1405
      1410
      1415
      1420
      1425
      1430
      1435
      1440
      1445
      1450
      1455
      1460
      1465
      1470
      1475
      1480
      1485
      1490
      1495
      1500
      1505
      1510
      1515
      1520
      1525
      1530
      1535
      1540
      1545
      1550
      1555
      1560
      1565
      1570
      1575
      1580
      1585
      1590
      1595
      1600
      1605
      1610
      1615
      1620
      1625
      1630
      1635
      1640
      1645
      1650
      1655
      1660
      1665
      1670
      1675
      1680
      1685
      1690
      1695
      1700
      1705
      1710
      1715
      1720
      1725
      1730
      1735
      1740
      1745
      1750
      1755
      1760
      1765
      1770
      1775
      1780
      1785
      1790
      1795
      1800
      1805
      1810
      1815
      1820
      1825
      1830
      1835
      1840
      1845
      1850
      1855
      1860
      1865
      1870
      1875
      1880
      1885
      1890
      1895
      1900
      1905
      1910
      1915
      1920
      1925
      1930
      1935
      1940
      1945
      1950
      1955
      1960
      1965
      1970
      1975
      1980
      1985
      1990
      1995
      2000
      2005
      2010
      2015
      2020
      2025
      2030
      2035
      2040
      2045
      2050
      2055
      2060
      2065
      2070
      2075
      2080
      2085
      2090
      2095
      2100
      2105
      2110
      2115
      2120
      2125
      2130
      2135
      2140
      2145
      2150
      2155
      2160
      2165
      2170
      2175
      2180
      2185
      2190
      2195
      2200
      2205
      2210
      2215
      2220
      2225
      2230
      2235
      2240
      2245
      2250
      2255
      2260
      2265
      2270
      2275
      2280
      2285
      2290
      2295
      2300
      2305
      2310
      2315
      2320
      2325
      2330
      2335
      2340
      2345
      2350
      2355
      2360
      2365
      2370
      2375
      2380
      2385
      2390
      2395
      2400
      2405
      2410
      2415
      2420
      2425
      2430
      2435
      2440
      2445
      2450
      2455
      2460
      2465
      2470
      2475
      2480
      2485
      2490
      2495
      2500
      2505
      2510
      2515
      2520
      2525
      2530
      2535
      2540
      2545
      2550
      2555
      2560
      2565
      2570
      2575
      2580
      2585
      2590
      2595
      2600
      2605
      2610
      2615
      2620
      2625
      2630
      2635
      2640
      2645
      2650
      2655
      2660
      2665
      2670
      2675
      2680
      2685
      2690
      2695
      2700
      2705
      2710
      2715
      2720
      2725
      2730
      2735
      2740
      2745
      2750
      2755
      2760
      2765
      2770
      2775
      2780
      2785
      2790
      2795
      2800
      2805
      2810
      2815
      2820
      2825
      2830
      2835
      2840
      2845
      2850
      2855
      2860
      2865
      2870
      2875
      2880
      2885
      2890
      2895
      2900
      2905
      2910
      2915
      2920
      2925
      2930
      2935
      2940
      2945
      2950
      2955
      2960
      2965
      2970
      2975
      2980
      2985
      2990
      2995
      3000
      3005
      3010
      3015
      3020
      3025
      3030
      3035
      3040
      3045
      3050
      3055
      3060
      3065
      3070
      3075
      3080
      3085
      3090
      3095
      3100
      3105
      3110
      3115
      3120
      3125
      3130
      3135
      3140
      3145
      3150
      3155
      3160
      3165
      3170
      3175
      3180
      3185
      3190
      3195
      3200
      3205
      3210
      3215
      3220
      3225
      3230
      3235
      3240
      3245
      3250
      3255
      3260
      3265
      3270
      3275
      3280
      3285
      3290
      3295
      3300
      3305
      3310
      3315
      3320
      3325
      3330
      3335
      3340
      3345
      3350
      3355
      3360
      3365
      3370
      3375
      3380
      3385
      3390
      3395
      3400
      3405
      3410
      3415
      3420
      3425
      3430
      3435
      3440
      3445
      3450
      3455
      3460
      3465
      3470
      3475
      3480
      3485
      3490
      3495
      3500
      3505
      3510
      3515
      3520
      3525
      3530
      3535
      3540
      3545
      3550
      3555
      3560
      3565
      3570
      3575
      3580
      3585
      3590
      3595
      3600
      3605
      3610
      3615
      3620
      3625
      3630
      3635
      3640
      3645
      3650
      3655
      3660
      3665
      3670
      3675
      3680
      3685
      3690
      3695
      3700
      3705
      3710
      3715
      3720
      3725
      3730
      3735
      3740
      3745
      3750
      3755
      3760
      3765
      3770
      3775
      3780
      3785
      3790
      3795
      3800
      3805
      3810
      3815
      3820
      3825
      3830
      3835
      3840
      3845
      3850
      3855
      3860
      3865
      3870
      3875
      3880
      3885
      3890
      3895
      3900
      3905
      3910
      3915
      3920
      3925
      3930
      3935
      3940
      3945
      3950
      3955
      3960
      3965
      3970
      3975
      3980
      3985
      3990
      3995
      4000
      4005
      4010
      4015
      4020
      4025
      4030
      4035
      4040
      4045
      4050
      4055
      4060
      4065
      4070
      4075
      4080
      4085
      4090
      4095
      4100
      4105
      4110
      4115
      4120
      4125
      4130
      4135
      4140
      4145
      4150
      4155
      4160
      4165
      4170
      4175
      4180
      4185
      4190
      4195
      4200
      4205
      4210
      4215
      4220
      4225
      4230
      4235
      4240
      4245
      4250
      4255
      4260
      4265
      4270
      4275
      4280
      4285
      4290
      4295
      4300
      4305
      4310
      4315
      4320
      4325
      4330
      4335
      4340
      4345
      4350
      4355
      4360
      4365
      4370
      4375
      4380
      4385
      4390
      4395
      4400
      4405
      4410
      4415
      4420
      4425
      4430
      4435
      4440
      4445
      4450
      4455
      4460
      4465
      4470
      4475
      4480
      4485
      4490
      4495
      4500
      4505
      4510
      4515
      4520
      4525
      4530
      4535
      4540
      4545
      4550
      4555
      4560
      4565
      4570
      4575
      4580
      4585
      4590
      4595
      4600
      4605
      4610
      4615
      4620
      4625
      4630
      4635
      4640
      4645
      4650
      4655
      4660
      4665
      4670
      4675
      4680
      4685
      4690
      4695
      4700
      4705
      4710
      4715
      4720
      4725
      4730
      4735
      4740
      4745
      4750
      4755
      4760
      4765
      4770
      4775
      4780
      4785
      4790
      4795
      4800
      4805
      4810
      4815
      4820
      4825
      4830
      4835
      4840
      4845
      4850
      4855
      4860
      4865
      4870
      4875
      4880
      4885
      4890
      4895
      4900
      4905
      4910
      4915
      4920
      4925
      4930
      4935
      4940
      4945
      4950
      4955
      4960
      4965
      4970
      4975
      4980
      4985
      4990
      4995
      5000
      5005
      5010
      5015
      5020
      5025
      5030
      5035
      5040
      5045
      5050
      5055
      5060
      5065
      5070
      5075
      5080
      5085
      5090
      5095
      5100
      5105
      5110
      5115
      5120
      5125
      5130
      5135
      5140
      5145
      5150
      5155
      5160
      5165
      5170
      5175
      5180
      5185
      5190
      5195
      5200
      5205
      5210
      5215
      5220
      5225
      5230
      5235
      5240
      5245
      5250
      5255
      5260
      5265
      5270
      5275
      5280
      5285
      5290
      5295
      5300
      5305
      5310
      5315
      5320
      5325
      5330
      5335
      5340
      5345
      5350
      5355
      5360
      5365
      5370
      5375
      5380
      5385
      5390
      5395
      5400
      5405
      5410
      5415
      5420
      5425
      5430
      5435
      5440
      5445
      5450
      5455
      5460
      5465
      5470
      5475
      5480
      5485
      5490
      5495
      5500
      5505
      5510
      5515
      5520
      5525
      5530
      5535
      5540
      5545
      5550
      5555
      5560
      5565
      5570
      5575
      5580
      5585
      5590
      5595
      5600
      5605
      5610
      5615
      5620
      5625
      5630
      5635
      5640
      5645
      5650
      5655
      5660
      5665
      5670
      5675
      5680
      5685
      5690
      5695
      5700
      5705
      5710
      5715
      5720
      5725
      5730
      5735
      5740
      5745
      5750
      5755
      5760
      5765
      5770
      5775
      5780
      5785
      5790
      5795
      5800
      5805
      5810
      5815
      5820
      5825
      5830
      5835
      5840
      5845
      5850
      5855
      5860
      5865
      5870
      5875
      5880
      5885
      5890
      5895
      5900
      5905
      5910
      5915
      5920
      5925
      5930
      5935
      5940
      5945
      5950
      5955
      5960
      5965
      5970
      5975
      5980
      5985
      5990
      5995
      6000
      6005
      6010
      6015
      6020
      6025
      6030
      6035
      6040
      6045
      6050
      6055
      6060
      6065
      6070
      6075
      6080
      6085
      6090
      6095
      6100
      6105
      6110
      6115
      6120
      6125
      6130
      6135
      6140
      6145
      6150
      6155
      6160
      6165
      6170
      6175
      6180
      6185
      6190
      6195
      6200
      6205
      6210
      6215
      6220
      6225
      6230
      6235
      6240
      6245
      6250
      6255
      6260
      6265
      6270
      6275
      6280
      6285
      6290
      6295
      6300
      6305
      6310
      6315
      6320
      6325
      6330
      6335
      6340
      6345
      6350
      6355
      6360
      6365
      6370
      6375
      6380
      6385
      6390
      6395
      6400
      6405
      6410
      6415
      6420
      6425
      6430
      6435
      6440
      6445
      6450
      6455
      6460
      6465
      6470
      6475
      6480
      6485
      6490
      6495
      6500
      6505
      6510
      6515
      6520
      6525
      6530
      6535
      6540
      6545
      6550
      6555
      6560
      6565
      6570
      6575
      6580
      6585
      6590
      6595
      6600
      6605
      6610
      6615
      6620
      6625
      6630
      6635
      6640
      6645
      6650
      6655
      6660
      6665
      6670
      6675
      6680
      6685
      6690
      6695
      6700
      6705
      6710
      6715
      6720
      6725
      6730
      6735
      6740
      6745
      6750
      6755
      6760
      6765
      6770
      6775
      6780
      6785
      6790
      6795
      6800
      6805
      6810
      6815
      6820
      6825
      6830
      6835
      6840
      6845
      6850
      6855
      6860
      6865
      6870
      6875
      6880
      6885
      6890
      6895
      6900
      6905
      6910
      6915
      6920
      6925
      6930
      6935
      6940
      6945
      6950
      6955
      6960
      6965
      6970
      6975
      6980
      6985
      6990
      6995
      7000
      7005
      7010
      7015
      7020
      7025
      7030
      7035
      7040
      7045
      7050
      7055
      7060
      7065
      7070
      7075
      7080
      7085
      7090
      7095
      7100
      7105
      7110
      7115
      7120
      7125
      7130
      7135
      7140
      7145
      7150
      7155
      7160
      7165
      7170
      7175
      7180
      7185
      7190
      7195
      7200
      7205
      7210
      7215
      7220
      7225
      7230
      7235
      7240
      7245
      7250
      7255
      7260
      7265
      7270
      7275
      7280
      7285
      7290
      7295
      7300
      7305
      7310
      7315
      7320
      7325
      7330
      7335
      7340
      7345
      7350
      7355
      7360
      7365
      7370
      7375
      7380
      7385
      7390
      7395
      7400
      7405
      7410
      7415
      7420
      7425
      7430
      7435
      7440
      7445
      7450
      7455
      7460
      7465
      7470
      7475
      7480
      7485
      7490
      7495
      7500
      7505
      7510
      7515
      7520
      7525
      7530
      7535
      7540
      7545
      7550
      7555
      7560
      7565
      7570
      7575
      7580
      7585
      7590
      7595
      7600
      7605
      7610
      7615
     
```

```

SUBROUTINE FOUR    74/74   OPT=1   PMDMP

      420    DD 530  I1=1,I1RNG,2
              DD 530  J3=1,I1,NON2,NP1
              KM1N=J3+IPAR*M
              1F (MMAX-NON2) 430,430,440
              KM1N=U3
              KM1F=I1PARDIMAX
              KSTEP=4*KDIF
              DO 520  K1=MMIN,NTOT,KSTEP
              K2=K1+KDIF
              K3=K2+KDIF
              K4=K3+KDIF
              1F (MMAX-NON2) 460,460,480
                  U1R=DATA(K1)+DATA(K2)
                  U1I=DATA(K1+1)+DATA(K2+1)
                  U2R=DATA(K3)+DATA(K4)
                  U2I=DATA(K3+1)+DATA(K4+1)
                  U3R=DATA(K1)-DATA(K2)
                  U3I=DATA(K1+1)-DATA(K2+1)
                  IF (ISIGN,470,475,475
                  U4R=DATA(K3+1)-DATA(K4+1)
                  U4I=DATA(K4)-DATA(K3)
                  GO TO 510
                  U4P=DATA(K4+1)-DATA(K3+1)
                  U4I=DATA(K3)-DATA(K4)
                  GO TO 510
                  12R=W2R*DATA(K2)-W2I*DATA(K2+
                  12I=W2R*DATA(K2+1)+W2I*DATA(K3+
                  T3R=WR*DATA(K3)-WI*DATA(K3+
                  T3I=WR*DATA(K3+1)+WI*DATA(K4-
                  T4R=W3R*DATA(K4)-WI*DATA(K4-
                  T4I=W3R*DATA(K4+1)+W3I*DATA(K4+
                  U1R=DATA(K1)+T2R
                  U1I=DATA(K1+1)+T2I
                  U2R=T3R+T4R
                  U2I=T3I+T4I
                  U3R=DATA(K1)-T2R
                  U3I=DATA(K1+1)-T2I
                  1F (ISIGN,470,500,500
                  U4R=T3I-T4I
                  U4I=T4R-T3R
                  GO TO 510
                  U4I=T4I-T3I
                  DATA(K1)=U1R+U2R
                  DATA(K1+1)=U1I+U2I
                  DATA(K2)=U1R-U2R
                  DATA(K2+1)=U3I-U4I
                  DATA(K3)=U3R-U4R
                  DATA(K3+1)=U1I-J2I
                  DATA(K4)=U3I-U4I
                  DATA(K4+1)=U1I-J3I

```

FTN 4, 8+528 03/21/81 12:15.51 PAGE 5

FTN 4.0+528 09/21/81 12.15.51 PAGE 6

SUBROUTINE FOURT	74/74	OPT=1	PMDMP
540	IF (ISIGN)540,550,550		
	TEMPR=WR		
	WR=-WI		
	WI=-TEMPR		
290	GO TO 560		
	TEMPR=WR		
550	TEMPR=WR		
	WR=WI		
	WI=TEMPR		
560	IF (M-LMAX)565,565,410		
565	TEMPR=WR		
	WR=WR*WSTPR-WI*WSTPI+WR		
570	WI=WI*WSTPR+TEMPR*WSTPI+WI		
	IIPAR=3-IIPAR		
	NMAX=NMAX-NMAX		
300	GO TO 360		
	C MAIN LOOP FOR FACTORS NOT EQUAL TO TWO. APPLY THE TWIDDLE FACTOR		
	W=EXP((ISIGN)*2*PI*SQRT((-1)*(J2-1)*(J1-1))/(NP2*IFP1)), THEN		
305	PERFORM A FOURIER TRANSFORM OF LENGTH IFACT(IF), MAKING USE OF		
	CONJUGATE SYMMETRIES.		
600	IF (INTNO-NP2)605,700,700		
605	IF P1=NON2		
	IF F=1		
	NP1:HF=NP1/2		
610	IFP2=IFP1/IFACT(IF)		
	J1RNG=NP2		
611	IF ((ICASE-3)16,611,612		
	J1RNG=(NP2+IFP1)/2		
	J2STP=NP2/IFACT(IF)		
	J1RG2=(J2STP+IFP2)/2		
612	J2MIN=1+IFP2		
	IF ((IFP1-NP2)16,640,640		
615	DO 635 J2=J2MIN,IFP1,IFP2		
	THETA=-TWOPI*FLOAT(J2-1)/FLOAT(NP2)		
320	IF (ISIGN)625,620,620		
	THETA=-THETA		
	SINTH=SIN(THETA/2.)		
	WSITPI=-2.*SINTH*SINTH		
	WSITPI=SIN(THETA)		
	WR=WSTPR+.		
625	WI=WSTPI		
	J1MIN=J2+IFP1		
325	DO 635 J1=J1MIN,J1RNG,IFP1		
	I1MAX=J1+I1RG-2		
330	DO 630 I1=J1,I1MAX,2		
	WI=TEMPR*WSTPR-WI*WSTPI+WI*WSTPR+WI		
335	DO 630 I3=I1,NTOT,NP2		
	J3MAX=I3+IFP2-NP1		
	DO 630 J3=3,J3MAX,NP1		
	IF (P1)=DATA(J3)		
	DATA(J3)=DATA(J3)+W1-DATA(J3+1)*W1		
	DATA(J3+1)=TEMPR,W1+DATA(J3+1)*W1		
	TEMPR=WR		
340	IF (P2)=WR		
	W1=TEMPR*WSTPR-WI*WSTPI+WI*WSTPR+WI		
625			

SUBROUTINE FOURT 74/74 DPT=1 PMDMP

FTN 4.8+528

PAGE 7

```

640  THETA=-TWOPI/FLOAT(IFACT(IF))          FFT T3320
     IF((SIGN)650,645,645      FFT T3330
645  THETA=-THETA      FFT T3340
650  SINTH=SIN(THETA/2.)      FFT T3350
     WSTPH=-2.*SINTH*SINTH
     WSTPI=SIN(THETA)
KSTEP=2,N/IFACT(IF)
KANG=KSTEP*(IFACT(IF)/2)+1
DO 698 11=1,1RNG,2
DO 698 13=L1 NTOT NP2
DO 690 KMIN=1,KANG,KSTEP
J1MAX=I3+J1RNG-1FP1
DO 680 J1=I3,J1MAX,1FP1
J3MAX=J1+1FP2-NP1
DO 680 J3=J1 J3MAX ,NP1
J2MAX=J3+1FP1-1FP2
K=KMIN+(J3-J1+(J1-13)/IFACT(IF))/NP1HF
IF(KMIN-1)655,655,665
655  SUMR=0.
SUMI=0.
DO 660 J2=J3,J2MAX,1FP2
     SUMR=SUMR+DATA(J2)
     SUMI=SUMI+DATA(J2+1)
WORK(K)=SUMR
WORK(K+1)=SUMI
GO TO 680
KCNJ=K+2*(N-KMIN+1)
J2=J2MAX
SUMR=DATA(J2)
SUMI=DATA(J2+1)
OLDSR=0.
OLDSI=0.
J2=J2-1FP2
670  TTEMPR=SUMR
TEMP1=SUMI
SUMR=TWOWR*SUMR-OLDSR+DATA(J2)
SUMI=TWOWR*SUMI-OLDSI+DATA(J2+1)
OLDSR=TEMPR
OLDSI=TEMP1
J2=J2-1FP2
1F ((J2-J3)675,675,670
TEMPR=wr*SUMR-OLDSR+DATA(J2)
TEMP1=wr*SUMI-OLDSI+DATA(J2+1)
TEMP1=WI*SUMR
WCRK(K+1)=TEMPR-TEMP1
WCRK(KCONJ+1)=TEMPR+TEMP1
TEMPR=wr*SUMI-OLDSI+DATA(J2+1)
TEMP1=WI*TEMPR+TEMP1
WCRK(K+1)=TEMPR+TEMP1
WORK(KCONJ+1)=TEMPR-TEMP1
CC;TRUE
JF((K+1)N-1)685,685,696
WR=wr*WSTPR-WI*WSTPI+WR
WR=wr*WSTPR-WI*WSTPR+WI
WI=TEMPR*WSTPI+WI*WSTPR+WI
385
675
380
385
390
680
685
395

```

SUBROUTINE	FOURT	74/74	OPT=1	PMDMP	FTN 4.8+528	09/21/81	12.15.51	PAGE
400	690	TWOWR=WR+WR IF (ICASE-3)692,691,692 IF (IFP1-NP2)695,692,692 K=1 12MAX=13+NP2-NP1 DO 693 I2=13,12MAX,NP1 DATA(I2)=WORK(K) DATA(I2+1)=WORK(K+1) K=K+2 GO TO 698						8
405	693	C COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N ODD, BY CON- C JUGATE SYMMETRIES AT EACH STAGE. C						
410	695	J3MAX=13+IFP2-NP1 DO 697 J3=13,J3MAX,NP1 J2MAX=J3+NP2-J2STP DO 697 J2=J3,J2MAX,J2STP J1MAX=J2+J1RG2-1FP2 J1CNJ=J3+J2MAX+J2STP-J2 DO 697 J1=J2,J1MAX,IFP2 K=1+J1-13 DATA(J1)=WORK(K) DATA(J1+1)=WORK(K+1) IF (J1-J2)697,697,696 DATA(J1CNJ)=WORK(K) DATA(J1CNJ+1)=WORK(K+1) J1CNJ=J1CNJ-1FP2 CONTINUE IF=IF+1 IFP1=IFP2 IF(IFP1-NP1)700,700,610						
420	697	C COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N EVEN, BY CON- C JUGATE SYMMETRIES. C						
425	698	700 GO TO (900,800,900,701),ICASE 701 NHALF=N						
430	699	N=N+1 THETA=-TWOPI/FLOAT(N) IFI ISIGN(703,702,702 THETA=-THETA SI*TH=SIN(THETA/2.) WSTPR=-2.*SINH*SI*TH WSTPI=SIN(THETA) WR=WSTPR+1. W1=WSTPI I*I=3 J*I=N-2*NHALF-1 GO TO 725						
435	700	702 THETA=						
440	701	703 SI*TH=SIN(THETA/2.) WSTPR=-2.*SINH*SI*TH WSTPI=SIN(THETA) WR=WSTPR+1. W1=WSTPI I*I=3 J*I=N-2*NHALF-1 GO TO 725						
445	702	704 SUMR=(DATA(I)+DATA(J))/2. SUMI=(DATA(I)+DATA(J+1))/2. DIFR=(DATA(I)-DATA(J))/2. CIRI=(DATA(I+1)-DATA(J+1))/2. TEMPR=WR*SUMI+WI*UIFR						
450	710	705						

SUBROUTINE FOURT 74/74 OPT=1 PNDMP FTN 4.8+528 09/21/81 12.15.51 PAGE 9

```

TEMP1=WI*SUM1-WR*DIFR
DATA(I)=SUMR+TEMP1
DATA(I+1)=DIFI+TEMP1
DATA(J)=SUMR-TEMPR
DATA(J+1)=-DIFI+TEMP1
J=J+NP2
I1IN=IMIN+2
JMIN=JMIN-2
TEMPR=WR
WR=WR*WSTPR-WI*WSTPI+WR
WI=TEMPR*WSTPI+WI*WSPR+WI
IF(I1IN-JMIN)710,730,740
725 I(F(1SIGN)731,740,740
730 DO 735 I=I1IN,NTOT,NP2
731 DATA(I+1)=-DATA(I+1)
735 DATA(I+1)=DATA(I+1)
740 NP2=NP2+NP2
NTOT=NTOT+NTOT
J=NTOT+1
IMAX=NTOT/2+1
IMIN=IMAX-2*NHALF
I=IMIN
GO TO 755
750 DATA(J)=DATA(I)
DATA(J+1)=-DATA(I+1)
755 I=I+2
J=J-2
IF(I-IMAX)750,760,760
760 DATA(J)=DATA(IMIN)-DATA(IMIN+1)
DATA(J+1)=0.
IF(I-J)770,780,780
765 DATA(J)=DATA(I)
DATA(J+1)=DATA(I+1)
770 I=I-2
J=J-2
IF(I-IMIN)775,775,765
775 DATA(J)=DATA(IMIN)+DATA(IMIN+1)
DATA(J+1)=0.
IMAX=IMIN
GO TO 745
780 DATA(1)=DATA(1)+DATA(2)
DATA(2)=0.
GO TO 900
C
C COMPLETE A REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION BY
C CONJUGATE SYMMETRIES.
C
800 IF(I1RNG-NP1)805,900,900
805 DO 860 I3=1,NTOT,NP2
12:MAX=13+NP2-NP1
DO 860 I2=13,12MAX,NP1
I1IN=12+I1RNG
12:MAX=I2+NP1-2
JMAX=2+I3+NP1-IMIN
IF(I2-13)920,820,B10
JMAX=JMAX+NP2
810 IF(ID1A-2)850,850,830
820 J=JMAX+NP0
830

```

SUBROUTINE FOUR:	74/74	OPT=1	PMDMP	FTN 4.8+528	09/21/81	12.15.51	PAGE	10	
515	840	I = IMIN, IMAX, 2 DATA(I)=DATA(J) DATA(I+1)=-DATA(J+1) J=J-2			FFT T5030 FFT T5040 FFT T5050 FFT T5060 FFT T5070 FFT T5080 FFT T5090 FFT T5100 FFT T5110 FFT T5120 FFT T5130 FFT T5140 FFT T5150 FFT T5160 FFT T5170 FFT T5180 FFT T5190				
520	850	J=JMAX DO 860 I = IMIN, IMAX, NPO DATA(I)=DATA(J) DATA(I+1)=-DATA(J+1) J=J-NPO							
525	860	C END OF LOOP ON EACH DIMENSION C NPO=NPI NP1=NP2 NPREV=N 910 920 RETURN END							
496	497	I DATA DATA	ARRAY ARRAY ARRAY	REFERENCE OUTSIDE DIMENSION BOUNDS. REFERENCE OUTSIDE DIMENSION BOUNDS. REFERENCE OUTSIDE DIMENSION BOUNDS.					
DIAGNOSIS OF PROBLEM									
SYMBOLIC REFERENCE MAP (R=1)									
ENTRY POINTS	3 FOURT	VARIABLES	SN	TYPE	RELOCATION	F.P.			
		0 DATA		REAL	ARRAY		1527 DIFI	REAL	
		1526 DIFR		REAL			1417 I	INTEGER	
		1416 ICASE		INTEGER			1403 IDIM	INTEGER	
		1412 IDIV		INTEGER			1411 IF	INTEGER	
		1532 IFACT		INTEGER			1475 IFP2	INTEGER	
		1473 IFP1		INTEGER			1475 IFP2	INTEGER	
		1530 IMAX		INTEGER			1524 IMIN	INTEGER	
		1435 IPAR		INTEGER			1413 IQUT	INTEGER	
		1414 IREM		INTEGER			1515 ISIGN	INTEGER	
		1427 I1		INTEGER			1426 I1MAX	INTEGER	
		1421 I1RIG		INTEGER			1425 I2	INTEGER	
		1521 I2MAX		INTEGER			1430 I3	INTEGER	
		1420 J		INTEGER			1521 JMAX	INTEGER	
		1525 JMIN		INTEGER			1505 J1	INTEGER	
		1522 J1C:J		INTEGER			1510 J1MAX	INTEGER	
		1504 J1MIN		INTEGER			1500 J1RG2	INTEGER	
		1476 J1RIG		INTEGER			1502 J2	INTEGER	
		1511 J2MAX		INTEGER			1501 J2MIN	INTEGER	
		1477 J2SIP		INTEGER			1431 J3	INTEGER	
		1506 J3MAX		INTEGER			1512 K	INTEGER	
		1475 JCP:J		INTEGER			1451 KDIF	INTEGER	

VARIABLES SN TYPE

		RELOCATION
1450	KMIN	INTEGER
1452	KSTEP	INTEGER
1437	K2	INTEGER
1454	K4	INTEGER
1441	LMAX	INTEGER
1440	NMAX	INTEGER
0	NDIN	INTEGER
0	NY	INTEGER
1434	NC42T	INTEGER
1422	NFO	INTEGER
1474	NP1HF	INTEGER
1424	NP2HF	INTEGER
1410	NTWO	INTEGER
1516	OLD5R	REAL
1514	SUM1	REAL
1433	TEMP1	REAL
1442	THETA	REAL
1520	TOD4R	REAL
1465	T2R	REAL
1467	T3R	REAL
1471	T4R	REAL
1455	U1R	REAL
1457	U2R	REAL
1461	U3R	REAL
1463	U4R	REAL
0	WORK	REAL
1400	WSTPI	REAL
1445	W2I	REAL
1447	W3I	REAL

EXTERNALS COS

TYPE

ARG5

REAL

1 LIBRARY

INLINE FUNCTIONS

TYPE

ARG5

REAL

1 INTRIN

STATEMENT LABELS

0 1

INACTIVE

0 2

INACTIVE

FTN 4.8+528

0 3

INACTIVE

0 4

INACTIVE

RELOCATI

0 5

INACTIVE

0 6

INACTIVE

OPT=1

0 7

INACTIVE

0 8

INACTIVE

PMDMP

0 9

INACTIVE

0 10

INACTIVE

RELOCATI

0 11

INACTIVE

0 12

INACTIVE

74/74

0 13

INACTIVE

0 14

INACTIVE

INTRIN

0 15

INACTIVE

0 16

INACTIVE

INACTI

0 17

INACTIVE

0 18

INACTIVE

INACTI

0 19

INACTIVE

0 20

INACTIVE

STATEMENT	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES	EXITS	EXT REFS	NOT INNER
520 600			0 605	6	INACTIVE	527	610	INACTIVE
0 611			542 612	6	INACTIVE	0	615	INACTIVE
0 620			555 625	6	INACTIVE	0	630	INACTIVE
0 635			637 640	6	INACTIVE	0	645	INACTIVE
645 650			655 0	6	INACTIVE	0	660	INACTIVE
725 665			737 670	6	INACTIVE	0	675	INACTIVE
771 690			0 685	6	INACTIVE	1005	686	INACTIVE
1014 690			0 691	6	INACTIVE	1024	692	INACTIVE
0 693			1043 695	6	INACTIVE	0	693	INACTIVE
1075 697			1106 698	6	INACTIVE	1120	700	
1131 701			0 702	6	INACTIVE	1137	703	
1153 710			0 720	6	INACTIVE	1214	725	
0 720			0 731	6	INACTIVE	0	735	
1230 740			1235 745	6	INACTIVE	1240	750	
1244 755			0 760	6	INACTIVE	1256	765	
1262 770			0 775	6	INACTIVE	1272	780	
1275 800			0 805	6	INACTIVE	0	810	
1315 820			0 820	6	INACTIVE	0	840	
1332 850			0 860	6	INACTIVE	1353	900	
0 910			1362 920	6				
LOOPS								
16 2	IDIM	89 91	68	INSTACK				
26 910	IDIM	96 528	13342	INSTACK				
116 80	J	159 161	23	INSTACK	NOT INNER			
136 150	J	172 189	433	INSTACK	NOT INNER			
144 125	J	175 183	21B	INSTACK	NOT INNER			
153 125	J	176 183	6B	INSTACK	NOT INNER			
212 340	J	201 210	26B	INSTACK	NOT INNER			
214 340	J	202 210	22B	INSTACK	NOT INNER			
222 340	K1	203 210	10B	OPT	NOT INNER			
267 570	L	222 297	22B	INSTACK	NOT INNER			
303 530	J1	229 284	1643	INSTACK	NOT INNER			
305 530	J3	230 284	16CB	INSTACK	NOT INNER			
332 520	K1	236 279	122B	OPT	EXT REFS	NOT INNER		
547 635	J2	319 342	7CB	INSTACK	NOT INNER			
571 635	J1	329 342	43B	INSTACK	NOT INNER			
575 630	J1	331 339	25B	INSTACK	NOT INNER			
577 630	J3	334 339	21B	OPT	EXT REFS	NOT INNER		
606 630	J3	336 339	6J	INSTACK	NOT INNER			
661 638	J1	351 428	233Y	INSTACK	NOT INNER			
663 698	J3	352 428	226B	INSTACK	NOT INNER			
664 610	KMIN	353 400	134B	INSTACK	NOT INNER			
670 6HJ	J1	355 392	107B	INSTACK	NOT INNER			
674 620	J3	357 392	10CB	INSTACK	NOT INNER			
715 660	J2	363 365	4B	INSTACK	NOT INNER			
1025 693	J2	405 408	45	INSTACK	NOT INNER			
1047 697	J3	415 427	37B	INSTACK	NOT INNER			
1053 697	J2	417 427	36B	INSTACK	NOT INNER			
1067 617	J1	420 427	10B	OPT	OPT			
1162 720	J1	451 462	17B	INSTACK	NOT INNER			
1225 735	J1	470 471	23	INSTACK	NOT INNER			
1301 810	J3	504 522	52B	INSTACK	NOT INNER			
1365 810	J2	506 522	43B	INSTACK	NOT INNER			
1326 840	J1	514 517	3B	INSTACK	NOT INNER			
1341 850	J1	519 522	33	INSTACK	NOT INNER			

STATISTICS	SUBROUTINE FOURT	74/74	OPT=1	PMDMP	FTN 4.8+528	09/21/81	12.15.51	PAGE 13
PROGRAM LENGTH	52000B CM USED	1662B	946					

SUBROUTINE GAUSS 74/74 OPT=1 PMDMP FTN 4.8+528 09/21/81 12.15.51 PAGE 1

```
1      C      SUBROUTINE GAUSS(IA,IY,VAL)
C      THIS ROUTINE CALCULATES A GAUSSIAN DISTRIBUTED RANDOM VARIABLE
C      VAL, WITH MEAN=0, AND STANDARD DEVIATION=1.
C
5      C      IA IS INITIALIZED BEFORE FIRST CALL TO ANY ODD INTEGER LESS THAN
C      10 DIGITS IN LENGTH.
C      IY IS GENERATED AND SHOULD BE USED FOR IA ON THE NEXT CALL TO
C      THIS ROUTINE.
C
     VAL=0.
10     DO 1 I=1,12
     X=RANF(DUM)
     CALL RANDOM(IA,IY,X)
     IA=IY
     VAL=VAL+X
     CONTINUE
     VAL=VAL-6.
     RETURN
     END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS	SN	TYPE	RELOCATION	24	I	INTEGER
3			*UNDEF		0	IY
GAUSS	26	REAL	F.P.		25	REAL
	0	INTEGER				INTEGER
	0	REAL				REAL
INLINE FUNCTIONS	TYPE	ARGS				
RANF	REAL	1	INTRIN			
STATEMENT LABELS						
0						
LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES	
12	1	I	10 15	6B	INSTACK	
STATISTICS						
PROGRAM LENGTH			338	27		
52000B CM USED						

SUBROUTINE RANDOM	74/74	OPT=1	PMDMP	FTN 4.8+528	09/21/81	12.15.51	PAGE 1
-------------------	-------	-------	-------	-------------	----------	----------	--------

```

1      C   SUBROUTINE RANDOM(IX,IY,YFL)
1      C   THIS ROUTINE GENERATES A UNIFORMLY DISTRIBUTED RANDOM VARIABLE,VAL,
1      C   WITH VALUE BETWEEN 0. AND 1.
1      C
5      C   IA IS SHOULD BE INITIALIZED BEFORE THE FIRST CALL TO THIS ROUTINE
5      C   TO AN ODD INTEGER LESS THAN 10 DIGITS IN LENGTH.
5      C   IY IS GENERATED BY THE PROGRAM AND SHOULD BE USED FOR IA ON THE NEXT
5      C   CALL TO THIS ROUTINE.
5      C
5      C   IY=IX*65539
5      C   IF(IY) 5,6,6
5      C   IY=IY+2147483647+1
5      C   YFL=IY
5      C   YFL=YFL*.4656613E-9
5      C   RETURN
5      C
10     C
10     C   END
10     C
15     C
15     C

```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 RANDOM

VARIABLES	SN	TYPE	RELOCATION	O	IY	INTEGER	F.P.
0 IX		INTEGER	F.P.				
0 YFL		REAL	F.P.				
STATEMENT LABELS							
0 S		INACTIVE	12	6			
STATISTICS							
PROGRAM LENGTH			20B	16			
52000B CM USED							

SUBROUTINE MULT

74/74 OPT=1 PMOMP

FTN 4.B+528

09/21/81

12.15.51

PAGE 1

```

1      SUBROUTINE MULT(A,B,L,M,N,C)
      REAL A(L,M),B(M,N),C(L,N)
      DO 300 I=1,L
      DO 200 J=1,N
      C(I,J)=0.
      DO 100 INDEX=1,M
      C(I,J)=C(I,J)+A(I,INDEX)*B(INDEX,J)
      CONTINUE
100   CONTINUE
200   CONTINUE
300   RETURN
      END

```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3 MULT

VARIABLES	SN	TYPE	RELOCATION	REAL	ARRAY	REAL	ARRAY	F.P.
0 A		REAL	F.P.	0	B	44	I	F.P.
0 C		REAL	F.P.			45	J	
46 INDEX		INTEGER				0	M	
0 L		INTEGER						
0 N		INTEGER						

STATEMENT LABELS			RELATION	REAL	ARRAY	REAL	ARRAY	F.P.
0 100			F.P.	0	B	44	I	F.P.
			F.P.			45	J	
			F.P.			0	M	

STATEMENT LABELS

0 100 0 200 0 300

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
12	300	I	3 10	278	NOT INNER
13	200	J	4 9	248	NOT INNER
30	100	INDEX	6 8	38	INSTACK

STATISTICS	PROGRAM LENGTH	57B	47
	52000B CM USED		

SUBROUTINE SPTN 74/74 OPT=1 PWDMP FTN 4.8+528
 1 SUBROUTINE SPTN(SIGMAB,R,M)
 DIMENSION R(64,64),C5)
 DATA C/.3679,.2431,.1353,.1069,.0591/
 5 C SET UP SPATIAL NOISE CORRELATION COEFFICIENT MATRIX
 C
 C C IS THE ARRAY CONTAINING THE NON-ZERO ELEMENTS
 C C CORRESPONDING TO THE DISTANCES TO NEIGHBORING PIXELS.
 C
 10 C SIGMAB IS THE BACKGROUND VARIANCE
 C
 C R IS THE M**2 BY M**2 CORRELATION MATRIX
 C
 15 C N=M**2
 DO 30 I=1,N
 DO 30 J=1,N
 R(I,J)=0.0
 CONTINUE
 DO 36 I=1,N
 R(I,I)=1.
 IF (I.GE.64) GO TO 36
 20 30 R(I,I+1)=C(1)
 IF (I.GE.63) GO TO 36
 R(I,I+2)=C(3)
 IF (I.GE.59) GO TO 36
 R(I,I+6)=C(4)
 IF (I.GE.58) GO TO 36
 R(I,I+7)=C(2)
 IF (I.GE.57) GO TO 36
 R(I,I+8)=C(1)
 IF (I.GE.56) GO TO 36
 R(I,I+9)=C(2)
 IF (I.GE.55) GO TO 36
 R(I,I+10)=C(4)
 IF (I.GE.51) GO TO 36
 R(I,I+14)=C(5)
 IF (I.GE.50) GO TO 36
 R(I,I+15)=C(4)
 IF (I.GE.49) GO TO 36
 R(I,I+16)=C(3)
 IF (I.GE.48) GO TO 36
 R(I,I+17)=C(4)
 IF (I.GE.47) GO TO 36
 R(I,I+18)=C(5)
 CONTINUE
 DO 37 I=1,M
 R(B*I-7,B*I)=0.0
 R(B*I-7,B*I-1)=0.0
 R(B*I-6,B*I)=0.0
 240 35 IF ((I.GE.3) GO TO 37
 R(B,I-8,I+1)=0.0
 R(B,I-8,I+2)=0.0
 R(B,I-1,B*I+1)=0.0
 R(B*I-7,B*I+7)=0.0
 R(B*I-7,B*I+8)=0.0
 40 45 36
 50 55

SUBROUTINE SPTN

74/74 OPT=1 ?MDMP

FTN 4.8+528

09/21/81 12.15.51

PAGE 2

```

R(8*I-6,8*I+8)=0.0
IF (I.GE.7) GO TO 37
R(8*I,8*I+9)=0.0
R(8*I,8*I+10)=0.0
R(8*I-1,8*I+9)=0.0
IF (I.GE.6) GO TO 37
R(8*I,8*I+17)=0.0
R(8*I,8*I+18)=0.0
R(8*I-1,8*I+17)=0.0
CONTINUE
DO 38 I=1,N
L=I+1
DO 38 J=L,N
IF (L.GT.N) GO TO 38
R(I,J)=R(I,J)
CONTINUE
DO 39 I=1,N
DO 39 J=1,N
R(I,J)=SIGMA B*R(I,J)
RETURN
END

```

```

70
38
75
39

```

```

      R(I,J)=SIGMA B*R(I,J)
      RETURN
END

```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3	SPTN	VARIABLES	SN	TYPE	RELOCATION	160	1	PROPERTIES
		163 C		REAL	ARRAY	162	L	INSTACK
		161 J		INTEGER		157	N	OPT
		0 M		INTEGER		0	SIGMAR	OPT
		0 R		REAL	ARRAY			REAL
		STATEMENT LABELS				66	36	
		0 30				0	39	
		133 38						
		LOOPS	LABEL	INDEX	FROM-TO	LENGTH	NOT INNER	
		11	30	I	17 20	133		
		16	30	J	18 20	28	INSTACK	
		12	26	I	21 47	3CB	OPT	
		13	37	I	48 67	173	OPT	
		117	36	I	68 73	213	INSTACK	NOT INNER
		131	38	J	70 73	48	INSTACK	NOT INNER
		141	39	I	74 76	148		
		146	39	J	75 76	3C	INSTACK	
		STATISTICS	PROGRAM LENGTH		176B	126		
			52000B CM USED					


```

60      IF(J-K) 62,65,62
62      KJ=IJ-I+K
       A(IJ)+A(IK)*A(KJ)+A(IJ)
       CONTINUE
       KJ=K-N
       DO 75 J=1,N
       KJ=KJ+N
       IF(J-K) 70,75,70
       A(KJ)=A(KJ)/BIGA
       CONTINUE
       D=D-BIGA
       A(KK)=1.0/BIGA
       CONTINUE
       K=N
       K=K-1
       IF(K) 150,150,105
       I=I(K)
       IF(I-K) 120,120,108
       JQ=N*(K-1)
       JR=N*(I-1)
       DO 110 J=1,N
       JK=JO+J
       HOLD=A(JK)
       JI=JR+J
       A(JK)=-A(JI)
       A(JI)=HOLD
       J=M(K)
       IF(J-K) 100,100,125
       DO 130 I=1,N
       KI=KI+N
       HOLD=A(KI)
       JI=KI-K+J
       A(KI)=-A(JI)
       A(JI)=HOLD
       GO TO 100
       DO 1002 I=1,NSQ
       SAVE=A(I)
       A(I)=B(I)
       B(I)=SAVE
       CONTINUE
       RETURN
       END
243
85      125      KI=K-N
                  DO 130 I=1,N
                  KI=KI+N
                  HOLD=A(KI)
                  JI=KI-K+J
                  A(KI)=-A(JI)
                  A(JI)=HOLD
                  GO TO 100
                  DO 1002 I=1,NSQ
                  SAVE=A(I)
                  A(I)=B(I)
                  B(I)=SAVE
                  CONTINUE
                  RETURN
                  END
95      130      150
1002

```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 INVERT

VARIABLES	SN	TYPE	REAL	ARRAY	F.P.
0 A		REAL	0 B	REAL	
306 BIGA		REAL	302 D	REAL	
313 HOLD		REAL	301 1	INTEGER	

VARIABLES	SN	TYPE	RELOCATION		
0 IER		INTEGER	F.P.	311 IJ	INTEGER
317 IK		INTEGER		310 12	INTEGER
3C7 J		INTEGER		314 JI	INTEGER
316 JK		INTEGER		315 JP	INTEGER
321 JQ		INTEGER		322 JR	INTEGER
304 K		INTEGER		312 KI	INTEGER
320 KJ		INTEGER		305 KK	INTEGER
324 L		REAL	ARRAY	524 M	REAL
0 N		INTEGER		303 NK	INTEGER
300 NSQ		INTEGER		323 SAVE	REAL
INLINE FUNCTIONS	TYPE	ARGS			
ABS	REAL	1 INTRIN			
STATEMENT LABELS					
0 10		INACTIVE		0 15	INACTIVE
0 25		INACTIVE		0 30	
0 38		INACTIVE		0 40	
0 46		INACTIVE		125 48	
137 55		INACTIVE		0 60	INACTIVE
165 65		INACTIVE		0 70	INACTIVE
0 80		INACTIVE		215 100	INACTIVE
0 108		INACTIVE		0 110	
0 125		INACTIVE		0 130	
0 1000		INACTIVE		0 1002	
LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
14 1000	I		6 7	2B	INSTACK
23 80	K		10 70	17CB	EXITS NOT INNER
32 20	J		16 24	26B	NOT INNER
44 20	I		18 24	1CB	OPT
72 30	I		28 33	5B	INSTACK
113 40	J		37 42	5B	INSTACK
133 55	I		47 51	5B	INSTACK
142 65	I		52 61	30B	NOT INNER
155 65	J		55 51	11B	OPT
177 75	J		63 67	5B	INSTACK
213 110	J		78 83	5B	INSTACK
254 130	I		87 92	5B	INSTACK
270 1002	I		94 98	3B	INSTACK

244

STATISTICS
PROGRAM LENGTH 756B
52000B CM USED 494

SUBROUTINE CHOLY 74/74 OPT=1 PMDMP

FTN 4.8+528

09/21/81

12.15.51

PAGE

1 SUBROUTINE CHOLY(A,N,S)
C THIS ROUTINE DETERMINES THE LOWER TRIANGULAR CHOLESKY SQUARE-
C ROOT OF AN NXN MATRIX.
C A IS THE INPUT MATRIX AND S IS THE CHOLESKY SQUARE-ROOT MATRIX.
5 DIMENSION A(N,N),S(N,N)
DO 1 I=1,N
DO 1 J=1,N
S(I,J)=0.
1 DO 123 I=1,N
DO 123 J=1,N
IF(ABS(A(I,J)).GT.1.E-06) GO TO 124
123 CONTINUE
RETURN
124 CONTINUE
S(1,1)=SQRT(A(1,1))
DO 5 I=2,N
IM1=I-1
DO 3 J=1,IM1
JM1=J-1
SUM=0.
DO 2 K=1,JM1
SUM=SUM+S(I,K)*S(J,K)
S(I,J)=(A(I,J)-SUM)/S(J,J)
SUM=0.
DO 4 K=1,IM1
SUM=SUM+S(I,K)**2
S(I,I)=SQRT(A(I,I)-SUM)
RETURN
END

15

20

25

30

35

40

45

50

55

60

65

70

75

80

85

90

95

100

105

110

115

120

125

130

135

140

145

150

155

160

165

170

175

180

185

190

195

200

205

210

215

220

225

230

235

240

245

250

255

260

265

270

275

280

285

290

295

300

305

310

315

320

325

330

335

340

345

350

355

360

365

370

375

380

385

390

395

400

405

410

415

420

425

430

435

440

445

450

455

460

465

470

475

480

485

490

495

500

505

510

515

520

525

530

535

540

545

550

555

560

565

570

575

580

585

590

595

600

605

610

615

620

625

630

635

640

645

650

655

660

665

670

675

680

685

690

695

700

705

710

715

720

725

730

735

740

745

750

755

760

765

770

775

780

785

790

795

800

805

810

815

820

825

830

835

840

845

850

855

860

865

870

875

880

885

890

895

900

905

910

915

920

925

930

935

940

945

950

955

960

965

970

975

980

985

990

995

1000

1005

1010

1015

1020

1025

1030

1035

1040

1045

1050

1055

1060

1065

1070

1075

1080

1085

1090

1095

1100

1105

1110

1115

1120

1125

1130

1135

1140

1145

1150

1155

1160

1165

1170

1175

1180

1185

1190

1195

1200

1205

1210

1215

1220

1225

1230

1235

1240

1245

1250

1255

1260

1265

1270

1275

1280

1285

1290

1295

1300

1305

1310

1315

1320

1325

1330

1335

1340

1345

1350

1355

1360

1365

1370

1375

1380

1385

1390

1395

1400

1405

1410

1415

1420

1425

1430

1435

1440

1445

1450

1455

1460

1465

1470

1475

1480

1485

18	- .32742E-01	.82705E-01	-.15966E+00	.10997E+00
19	- .69165E-01	.11001E+00	-.15362E+00	.88684E-01
20	- .96786E-01	.99853E-01	-.15902E+00	.92857E-01
FRAME	CNER (-)	SCNER (-)	YCER (-)	SYCER (-)
1	0.	0.	0.	0.
2	- .69820E-01	.55503E-01	-.74382E-01	.78520E-01
3	- .74801E-01	.67689E-01	-.14037E+00	.95165E-01
4	- .11632E+00	.75570E-01	-.12402E+00	.92589E-01
5	- .51652E-01	.62306E-01	-.17440E+00	.11644E+00
6	- .10002E+00	.10359E+00	-.16278E+00	.10724E+00
7	- .12020E+00	.10436E+00	-.19035E+00	.89767E-01
8	- .10405E+00	.10507E+00	-.15933E+00	.63890E-01
9	- .10955E+00	.67413E-01	-.15591E+00	.90686E-01
10	- .84575E-01	.96341E-01	-.19815E+00	.92536E-01
11	- .12637E+00	.76128E-01	-.16110E+00	.76475E-01
12	- .10394E+00	.92007E-01	-.20185E+00	.98832E-01
13	- .14933E+00	.98687E-01	-.19647E+00	.82057E-01
14	- .13257E+00	.11088E+00	-.22185E+00	.87295E-01
15	- .16291E+00	.10401E+00	-.21630E+00	.78569E-01
16	- .15397E+00	.91846E-01	-.24017E+00	.92153E-01
17	- .12503E+00	.79303E-01	-.22507E+00	.10160E+00
18	- .16037E+00	.86538E-01	-.21018E+00	.10280E+00
19	- .10670E+00	.64836E-01	-.24397E+00	.85960E-01
20	- .12566E+00	.10043E+00	-.22881E+00	.76848E-01
FRAME	CNER (+)	SCNER (+)	YCER (+)	SYCER (+)
1	0.	0.	0.	0.
2	- .16206E-01	.37668E-01	-.49023E-01	.53220E-01
3	- .15073E-01	.45831E-01	-.88350E-01	.59031E-01
4	- .25533E-01	.45228E-01	-.96539E-01	.51950E-01
5	- .19174E-01	.59691E-01	-.10369E+00	.58003E-01
6	- .25940E-01	.48264E-01	-.10165E+00	.47441E-01
7	- .27335E-01	.57338E-01	-.11587E+00	.43245E-01
8	- .27682E-01	.50440E-01	-.10571E+00	.50410E-01
9	- .11064E-01	.51635E-01	-.12203E+00	.59952E-01
10	- .23037E-01	.55718E-01	-.128332E+00	.40686E-01
11	- .29029E-01	.41689E-01	-.13329E+00	.52802E-01
12	- .28087E-01	.56174E-01	-.14258E+00	.59550E-01
13	- .29211E-01	.47210E-01	-.14630E+00	.42024E-01
14	- .40600E-01	.59072E-01	-.14767E+00	.59739E-01
15	- .27047E-01	.49159E-01	-.15651E+00	.43053E-01
16	- .33019E-01	.46264E-01	-.16217E+00	.49056E-01
17	- .33242E-01	.51401E-01	-.15956E+00	.57207E-01
18	- .30152E-01	.24009E-01	-.15995E+00	.61265E-01
19	- .20596E-01	.40643E-01	-.17455E+00	.61010E-01
20	- .28160E-01	.56560E-01	-.17086E+00	.59352E-01

RUNS=20
GAUSSIAN COVARIANCE 2.00

FRAMES=20
BACKGROUND VARIANCE= 1.0
NUMBER FREQ ZEROED= 2

NUMBER ZERO PAD=8
SMOOTHING ALPHA=.100
TRUTH MODEL UNCERTAINTY=.10

FRAME	MEAN ERROR OF ERROR IN H	MEAN VARIANCE IN D/DX	MEAN ERROR IN D/DX	MEAN VARIANCE OF ERROR IN D/DX	MEAN ERROR IN D/DY	MEAN VARIANCE OF ERROR IN D/DY
1	-.63852E-01	.77407E+00	.29364E+00	.91872E+00	.59740E+00	.94072E+00
2	-.78734E-01	.45121E+00	.29909E+00	.52321E+00	.611128E+00	.520088E+00
3	-.65521E-01	.31401E+00	.28246E+00	.35620E+00	.36250E+00	.48295E+00
4	-.65697E-01	.25642E+00	.26526E+00	.29197E+00	.45717E+00	.302226E+00
5	-.66817E-01	.22849E+00	.25812E+00	.25834E+00	.43446E+00	.26452E+00
6	-.62624E-01	.20263E+00	.24929E+00	.22169E+00	.42009E+00	.22908E+00
7	-.63430E-01	.17133E+00	.25264E+00	.19121E+00	.399912E+00	.19376E+00
8	-.62613E-01	.14969E+00	.25402E+00	.16766E+00	.16679E+00	.16679E+00
9	-.60624E-01	.13633E+00	.25443E+00	.15475E+00	.15176E+00	.15176E+00
10	-.44639E-01	.12953E+00	.26013E+00	.14437E+00	.14481E+00	.14481E+00
11	-.47623E-01	.12221E+00	.26158E+00	.13393E+00	.13335E+00	.13335E+00
12	-.44049E-01	.11599E+00	.26573E+00	.12471E+00	.12493E+00	.12493E+00
13	-.43781E-01	.11306E+00	.27122E+00	.11574E+00	.11513E+00	.11513E+00
14	-.44153E-01	.10619E+00	.26550E+00	.10832E+00	.10514E+00	.10514E+00
15	-.50219E-01	.10563E+00	.27118E+00	.10359E+00	.10049E+00	.10049E+00
16	-.46696E-01	.99767E-01	.27215E+00	.97111E-01	.97181E-01	.97181E-01
17	-.48134E-01	.97429E-01	.27263E+00	.91025E-01	.92946E-01	.92946E-01
18	-.55025E-01	.95342E-01	.28101E+00	.89119E-01	.90004E-01	.90004E-01
19	-.63420E-01	.93616E-01	.28334E+00	.86318E-01	.89903E-01	.89903E-01
20	-.66795E-01	.92066E-01	.27789E+00	.84924E-01	.85924E-01	.85924E-01

GAUSSIAN TARGET COVARIANCE VALUE

CSB NOS/BE L530C L530C-CMR3 07/13/81
12.15.38.DRM1VCP FROM CSA/1V
12.15.33.1P 00010944 WORDS - FILE INPUT , DC 04
12.15.38.DRM,T397,1090,CM150000,A720001, RM#
12.15.38.118 MCGREW ASD/ASD
12.15.44.ATTACH,IMSL, ID=LIBRARY, SN=ASD.
12.15.44.PFN IS
12.15.44.1:ISL
12.15.44.AT CY= 999 SN=ASD
12.15.44.LIBRARY,IMSL.
12.15.48.FTN,PWD.
12.18.50. 13.426 CP SECONDS COMPIRATION TIME
12.18.50.LGO.
13.04.55. STOP
13.04.55. 133500 MAXIMUM EXECUTION FL.
13.04.55. 260.757 CP SECONDS EXECUTION TIME.
13.04.55.OP 00025728 WORDS - FILE OUTPUT , DC 40
13.04.55.RS 29184 WORDS (51072 MAX USED)
13.04.55.CPA 295.343 SEC. 147.872 ADJ.
13.04.55.ID 42.106 SEC. 23.832 ADJ.
13.04.55.CM 14116.512 KWS. 114.621 ADJ.
13.04.55.CRUS 286.326
13.04.55.COST \$ 13.06
13.04.55.PP 69.942 SEC. DATE 09/21/81
13.04.55.EU END OF JOB, 1V A720001.

Correlator-Kalman Filter
Modcomp Software


```

1      PHAL(AN) =CC(FIL
2      REAL YMAX(4),Y(17),CNPE(4),YMAX(3),Y(64,64)
3      REAL Z1(10)
4      REAL F1(10,2)
5      REAL X(7,64),Y(64,8),CNPE(10,8),Y(2,8),Y(2,11)
6      REAL X(64)*Y(64),UC(7,76)
7      INTGRL AN(2)
8      COMPLEX DATA(24,24),ICLK(50),SAVE(24,24),DX(24,24),DY(24,24)
9      COMPLEX SDATA(24,24)

10     C
11      DATA STRUCTURES TO GATHER STATISTICS ON FILTER
12      YF1F2 IS THE LENGTH BETWEEN THE PREDICTED X DYNAMIC LOCATION
13      AT A PARTICULER STATE TIME AND THE TRUTH MODEL TRUE
14      X DYNAMIC LOCATION
15      YF1F2 IS THE SQUARE OF THE XF1X
16      YF1F2 IS THE LENGTH BETWEEN THE PREDICTED AND MEASURED X
17      TRACKED CAPABILITY
18      YF1F2 IS THE LENGTH BETWEEN THE FIRST ROW IN EACH IS USED FOR THE X
19      DIRECTION WHILE THE SECOND ROW IS FOR THE Y DIREC
20      CNPF IS THE ERROR IN THE PREDICTED LOCATION OF THE CENTROID
21      AT A PARTICULAR STATE TIME COMPARED TO THE TRUTH MODEL
22      CNPF IS THE SQUARE OF CNPF
23      CNPF IS THE LENGTH OF CNPF
24      CNPF IS THE LENGTH OF CNPF
25      ACTE AGAIN THE DIMENSION OF CNPF AND CNMFC2
26      REAL CNPF(2,20),CNPF(2,20),CNMFC(2,20),CNMFC(2,20)
27
28      CNPF IS THE ERROR BETWEEN THE UPDATED DYNAMIC LOCATION AT A
29      PARTICULAR PLUS TIME AND THE TRUTH MODEL TRUE DYNAM
30      XF1F2 IS THE SQUARE OF XF1
31      XF1F2 IS THE LENGTH OF XF1
32      NOTE THE DIMENSIONALITY OF XF1F2 FOR THE SAME REASONS AS
33      ACTE AGAIN THE 20 ALLOWS COMPUTATION OF STATISTICS PER
34      ACTE THE 20 ALLOWS COMPUTATION OF STATISTICS PER
35      ACTE
36      CNPF IS THE CENTROID ERROR AT THE PLUS TIME
37      CNPF IS THE CENTROID ERROR AT THE PLUS TIME
38      CNPF IS THE CENTROID ERROR AT THE PLUS TIME
39      CNPF IS THE CENTROID ERROR AT THE PLUS TIME
40      CNPF IS THE CENTROID ERROR AT THE PLUS TIME
41      FILTERS DATA STRUCTURES
42      PHIR IS THE STATE TRANSITION MATRIX FOR THE KALMAN FILTER
43      SIEVE SUBROUTINE FILTER
44      CFD IS THE RESULT OF THE INTEGRAL TERM IN THE PROPAGATION
45      OF THE COV MATRIX SEE SUBROUTINE PROPF
46      FFER IS THE FILTERS COVARIANCE MATRIX PLUS AFTER INCORPORAT
47      OF A MEASUREMENT
48      PFN IS THE FILTERS COVARIANCE MATRIX MINUS AFTER PROPAGATI
49      -HUT FILTER TO MEASUREMENT INCORPORATION
50      XFP IS THE FILTER STATE VECTOR PLUS
51      XFM IS THE FILTER STATE VECTOR PLUS
52      XFM IS THE FILTER STATE VECTOR PLUS
53      PHIF(4,4)=CFD(4,4)+PFF(4,4),PFF(4,4)=XFM(4)*XFH(4)
54
55

```

CLASSIC FUNCTIONS IN D.1 09-29-t1 39:13 HCOFIL PAGE 2

```
56 C 2 IS THE KALMAN FILTER MEASUREMENT VECTOR
57 C
58 C REAL Z(64)
59 C
60 C DATA M/24.24/
61 C
62 C INITIALIZE THE FILTERS DATA STRUCTURES
63 C
64 C DATA M/0.03233537/
65 C DATA INF/1.5/
66 C DEPIK,BFILE/ACQ416.DAT,0.03528/
67 C
68 C.....INITIALIZATION.....*
69 C.....INITIALIZATION.....*
70 C.....INITIALIZATION.....*
71 C
72 C 54121 INITIATE 3717)
73 C 3787 FCN416(X,X,GAUSSIAN TARGET COVARIANCE VALUE)
74 C REFERENCE:6.6.5.1,FID=6.21) COV
75 CFCN416(X,X)
76 CFCN416(X,X)
77 C 3701 FCN416(X,X,NUMBER OF ZERGES TO PAD)
78 C 54121(FIELD) NZ
79 CFCN416(X,X)
80 NZ=2*-NZ
81 C
82 C 3782 FORMAT(1X,*INPUT OF FRAME*)
83 CREFILE=5610) REFILLS
84 C
85 C 4023 FOR A CITY *SUPPLY OF SIMULATIONS*
86 CREFILE=5611) NRNGE
87 CWFITE(6,5763)
88 C 3793 FOR A CITY *ALPHA FOR SMOOTHING*
89 CREFILE=5610) ALPHA
90 C
91 C 3784 WFITE(6,5763)
92 CREFILE=5611) WFRCL
93 C
94 C 3785 FCN416(X,X,UPPER OF HIGH FREQ COMPONENTS TO ZERO*)
95 C
96 C 3789 FCN416(X,X,MEASUREMENT ERROR_VARIANCE*)
97 CREAD(E,2) VARM
98 CFORMAT(E,2)
99 C
100 C CALL INITF(TAF,VARDF,VARAF)
101 C
102 C SELLIE TILT TARGET AS 3 INDEPENDENT GAUSSIAN FUNCTIONS WITH
103 C VARIANCE=CCV.
104 C
105 C S(1)=1./COV
106 C S(7)=C.
107 C S(5)=C.
108 C S(6)=S(1)
109 C S(4)=S(1)
110 C S(3)=C.
```

```

111      S(7)=0.
112      S(10)=S(1)
113      S(1)=S(1)
114      S(11)=S(6)
115      S(12)=S(5)
116      S(13)=S(11)

117      C INITIALIZE TARGET INTENSITY ASSUMING
118      C 3 CIRCULAR CROSS-SECTION GAUSSIAN TARGET.
119      C
120      C
121      I=MAX(1)=20.
122      IPAY(2)=20.

123      IMAX(3)=20
124      C DEFINE TRUTH MOLL DYNAMICS
125      C
126      UPINT(1)=9.965
127      C-----ESTIMATES OF ID. DEV OF TRUTH MODEL ATMOSPHERIC JITTER.
128      C-----AFTER A SIGOT
129      CALL TRUTHCHIT,RCRROT,H,SIGOT,UT
130      C
131      C INITIALIZE THE FILTERS PARAMETERS
132      C
133      C-----INITIALIZE THE FILTERS MATRICES DEFINITION
134      C
135      CALL FILTER(TDF,VARDF,LAF,VARAF,UT,PHIF,QFD)
136      C
137      C INITIALIZE THE FILTER EARGE MATRICES TO ZERO
138      C
139      DO 21 J=1,12
140      CC(21,J)=0.20
141      XFTL(1,J)=0
142      XFTL(21,J)=0
143      CRWF(1,J)=0.
144      CRWF(21,J)=0.
145      CFEC(1,J)=C
146      CFEC(21,J)=C
147      CRPC(1,J)=C
148      CRPC(21,J)=C
21      CALL SPTRVAFM,R,F
149      C USING FIRST AND SECOND NEAREST NEIGHBOR DETERMINE THE CHOLESKY
150      C SQUAREGT. F. OF THE MEASUREMENT COVARIANCE MATRIX. R
151      C
152      C
153      C-----THIS LOGIC MAKES SPATIALLY CORRELATED/UNCORRELATED NCISF
154      C-----CONTINUE THE NEXT FOUR LINES IF WANT SPATIAL CORRELATION
155      C
156      DO 642 t l=1,64
157      DO 640 J=1,64
158      C-----N(1,J)=6.
159      C-----IF(I.EQ.J) R(I,J)=VAPM
160      642R CONTINUE
161      C-----FOR COMP VERSION OF CHOLESKY PUTS ROOT BACK INTO CALLING MATRIX
162      C-----CALL CHOLESKY(R,t4)
163      C-----F N D . I N I T I A L I Z A T I O N
164      C-----C
165      C-----C

```

```
146 C
147 C***** EIGHT MONTE-CARLO SIMULATION ****
148 C
149 C DRAFT VERSIONS OF AFRAMES EACH FOR MONTE-CARLO ANALYSIS
150 C
151 C TO DO NS=1, NFRAMES
152 C
153 C NSMFT=0.
154 C YSMFT=0.
155 C
156 C INITIALIZE SCATTERED DATA ARRAY
157 C
158 C DO 7 I=1,24
159 C      70 7 J=1,24
160 C      70 7 DATA(I,J)=CMLX(0.,0.)
161 C
162 C
163 C INITIALIZE STATE VECTOR
164 C
165 C      71 71 L=1,6
166 C      71 71 YT(L,1)=0.
167 C      71 71 XT(L,1)=0.5,0.
168 C      71 71 YT(2,1)=2.,0.
169 C      71 71 YT(3,1)=0.,0.
170 C      71 71 YT(4,1)=3.,0.
171 C
172 C
173 C
174 C
175 C
176 C
177 C INITIALIZE SCATTERED DATA ARRAY
178 C
179 C DO 7 I=1,24
180 C      70 7 J=1,24
181 C      70 7 DATA(I,J)=CMLX(0.,0.)
182 C
183 C INITIALIZE STATE VECTOR
184 C
185 C      71 71 L=1,6
186 C      71 71 YT(L,1)=0.
187 C      71 71 XT(L,1)=0.5,0.
188 C      71 71 YT(2,1)=2.,0.
189 C      71 71 YT(3,1)=0.,0.
190 C      71 71 YT(4,1)=3.,0.
191 C
192 C
193 C
194 C
195 C
196 C
197 C
198 C
199 C
200 C
201 C
202 C
203 C
204 C
205 C
206 C
207 C
208 C
209 C
210 C
211 C
212 C
213 C
214 C
215 C
216 C
217 C
218 C
219 C
220 C
```

TF=1K TARGET FOR AFRAME FRAMES (TIME SLICES)

DO 50 NR=1,NFRAMES

LFFINE GAUSSIAN PEAK LOCATIONS BASED ON COTHOID POSITION, YT

```
221 YMAX(1)=YT(1,1)
222 YMAX(2)=YT(1,1)-2.
223 YMAX(3)=YT(1,1)+2.
224 YMAX(1)=YT(2,1)-5.*1.666666
225 YMAX(2)=YT(2,1)+5.*1.666666
226 YMAX(3)=YT(2,1)+1.333333
227 C GET MEASUREMENT N.ISE ARRAY
228 C
229 C CALL POISS(4,0.64)
230 C CALL PULTE(XW,64,0.64,1,V)
231 C ----- GET MEASUREMENT DATA
232 C
233 C CALL TOTAL(CIMAX,S*XFAX,YMAX,X,Y*DATA,DX,DY)
234 C ADD CORRELATED NOISE TO CENTER EX8 PIXEL DATA
235 C
236 C
237 C DO 4 I=1,6
238 C   DO 4 J=1,6
239 C     DATA(I+J,e)=DATA(I+J,J+8)+CMPLX(V*(I-1)+J),0.0)
240 C   CG,TIP,UE
241 C
242 C AND UNCORRELATED NOISE TO MEASUREMENT DATA OUTSIDE CENTER
243 C EX8 PIXEL AREA
244 C
245 C CALL NOISE(UC,576)
246 C DO 6 J=1,24
247 C   DO 7 J=1,24
248 C     IF(I.GT.6.AND.I.LE.16.AND.J.GE.9.AND.J.LE.16) GO TO 6
249 C     IF(I.LT.7).OR.(J.LE.92).OR.(J.GE.17) GO TO 6
250 C     DATA(I,J)=DATA(I,J)+CMPLX((UC(24*(I-1)+J,0)).*SQR(T(UC)))
251 C
252 C CONTINUE
253 C
254 C CREATE THE MEASUREMENT VECTOR FOR THE FILTER UPDATE
255 C
256 C   DO 101 I=6,16
257 C     DO 101 J=4,16
258 C       K=K+1
259 C       Z(K)=REAL(DATA(I,J))
260 C
261 C       IF(K.NE.1) CALL CORREL(CNN,DATA,S,DATA,XCENT,YCENT,X,Y)
262 C       Z(1)=EXCIT
263 C       Z(2)=YCFIT
264 C
265 C       GO CALCULATE THE ERRORS OF THE FILTERS & ESTIMATE PRIOR TO
266 C       MEASUREMENT INCORPORATION
267 C       CALL STATFM(XFNE,XFRE2,CNRE,CNN,XI,YI,NR)
268 C
269 C       INCORPORATE MEASUREMENT
270 C
271 C       IF(NP.EQ.1) GO TO 164
272 C       CALL UPCKT(21,XFP,PFM,FIL)
273 C       CALCULATE THE ERRORS FOR THE FILTER AFTER INCORPORATION
274 C       OF THE MEASUREMENT
275 C       CALL STATFM(XFPP,XFNE2,CNPE,CNPZ,XTP,YT,XFP,NR)
```

```

      C COMPUTE THE SHIFT INFORMATION FROM THE CENTER OF FOV
  277 C XSHIFT=X-FP(1)+4.-XF(3)
  278 C YSHIFT=Y-FP(2)+4.-XF(4)
  279 C
  280 C SHIFT THE DATA ARRAY APPROPRIATELY
  281 C
  282 C GET FORWARD FF
  283 C
  284 C 164 CALL FOURT(DATA•NN•2,-1•1•WORK)
  285 C FILTER DESIRED FREQUENCY COMPONENTS OUT
  286 C
  287 C IF(NFREQ.GT.12) NFREQ=12
  288 C IF(NFREQ.LE.0) GO TO 3796
  289 DO 8 I=SF•IEF
  290   DC 3 JE 124
  291 C DATA(I,J)=CMPLX(0.,0.)
  292 C DATA(J,I)=CMPLX(0.,0.)
  293 R CONTINUE
  294 C
  295 3796 CONTINUE
  296 C
  297 C ASSUME IF I=0 THAT THE DATA IS CENTERED
  298 I=CM(1,1) CALL SHIFTDATA,24,XSHIFT,YSHIFT
  299 CALL SMOOTHDATA,SQDATA,ALPHA,24,NR
  300 CALL PROPF(PHIF,GFD,PPM,PFM,XFP,XFM)
  301 XEXFM(1)=4.
  302 Y=XFY(2)-4.
  303 C PROPAGATE TRUTH MODEL STATE ONE FRAME
  304 CALL SMOOTH1,GRD001B,XI,YI,E,2)
  305 C
  306 70 CONTINUE
  307 C
  308 C** END MONTE CARLO SIMULATION
  309 C
  310 C
  311 C CALCULATE MEAN AND VARIANCE STATISTICS
  312 C
  313 CALL FILST(XFME•XFME2•CNME•CNME2•XFPE•XFPE2•CNFE•CNPE2•NRUNS,
  314 #NFRAMES)
  315 C
  316 WRITE(6,9987) NRUNS,NFRAMES,NZ,NFREQ,COV,VARM,ALPHA,
  317 F SIGCT
  318 9987 FORMAT(1H1,T10•RHS=•,12,T13•FRAMES=•,12,T13•NUMBER.ZERO.PAD=•
  319   T1,T10•NUMBER.FREQ.ZERO=•,12,T10•GAUSSIAN.COVARIANCE=•
  320   FS•2,T28•MEASUREMENT.NOISE.VARIANCE=•,FS•1,T13•SMOOTHING.ALPHA
  321   =•,FT•3,T10•
  322 R *ISIGHT MODEL.UNCERTAINITY=•,E7,3,///)
  323 G0 10 54321
  324 6421 STOP
  325 END

```

```

1      SUBROUTINE UFCCMF(Z1,XFF,YFM,PFM,RFIL)
2      REAL Z1(4),XFF(4),YFM(4),YF(4,4),PFM(4,4),RFIL(2,2)
3      REAL TBL1(4,2),TSP2(2,2),THINV(2,2),TEMP3(2,4),TEMP4(4,4)
4      REAL HT(4,2),H(2,4),GACCR(4,2)
5      IFC=1,I=1,J=1
6      H(1,1)=1.
7      H(1,2)=0.
8      H(2,1)=0.
9      H(2,2)=1.
10     H(3,1)=0.
11     H(3,2)=0.
12     H(4,1)=0.
13     H(4,2)=1.
14     DO 4 J=1,2
15     H(I,J)=HT(I,J)
16     CALL MULT(PFM,H1,4,4,2,1,SP1)
17     CALL MULT(YFM,TBL1,2,4,2,1,HT2)
18     NC=1,I=1,J=1
19     NC=1,I=1,J=2
20     1   TEMP2(I,J)=TEMP2(I,J)+RFIL(I,J)
21     CALL MULT(YFM,2,1,HT1,V)
22     CALL MULT(YFM,2,2,1,HT1)
23     CALL MULT(YFM,TEMP1,4,4,2,GACCR)
24     CALL PFM(PFM,HT,2,4,4,BSP3)
25     CALL MULT(GACCR,TEMP3,4,2,4,TEMP4)
26     NC=2,I=1,J=4
27     DO 2 J=1,4
28     2   PFM(I,J)=PFM(I,J)-TEMP4(I,J)
29     CALL MULT(YFM,2,4,4,1,RES1)
30     ON 3 I=1,2
31     3   RES1(I)=714(I)-RES1(I)
32     CALL MULT(GACCR,RES1,4,2,1,UPC)
33     DO 4 I=1,4
34     4   YFR(I)=XFRC(I)*UPC(I)
35     RETURN
      END

```

```

1      SUBROUTINE C2REFL(XN,DATA,TEMPLATE,XCENT,YCENT,X,Y)
2      CALL TBLAT(24,74),DATA(24,24)
3      CALL TBLP(24,74)
4      CALL TBLS(24,74)
5      CALL TBLL(24,74)
6      DATA(1) = P(1)
7      CALL FRUITCATA(XN,2,-1,1,LCRK)
8      P(1) = 1.24
9      L(1,J)=1,24
10     TBL(1,J)=C1,J=1,24,FLATE(1,J),DATA(1,J)
11     CALL FEATCATA(XN,2,1,1,LCRK)
12     CALL FEATCERP(XN,2,1,1,LCRK)
13     L(0,J)=1,24
14     L(0,J)=1,24
15     TBL(1,J)=TBL(1,J)/576.
16     DATA(1,J)=DATA(1,J)/576.
17     CALL LBLAY(24,1,DATA)
18     CALL CHANG(1,24)
19     CALL CBLAY(24,24,DATA)
20     P(0,J)=1.24
21     L(0,J)=1,24
22     DATA(1,J)=FAL(1,J)
23     L(1,J)=112) ((DATA(1,J),J=1,24),I=1,24)
24     I12 = FORMAT(1,5,0,12)
25     CALL CHANLIC(XN,Y1LP,24,XCENT)
26     CALL DISPLAY(24,74,DATA)
27     XCET=XCET-5
28     YCENT=YCENT-5
29     WRITE(1,110) XCENT,YCENT
30     110   FORMAT(1X,'CENTRID=(0,F7.2,0,0,F7.2,0,0)')
31     C     >CX=CUM*(XCENT-XSHIFT)
32     C     YCUM=YCUM*(YCNT-YSHIFT)
33     C     XCUS=YCUM*(XCENT-XSHIFT)**2
34     C     YCUM2=YCUM2*(YCET-J-YSHIFT)**2
35     1000  CONTINUE
36     CALL STATEYCUM,YCUM,XCUM2,YCUM2,N1)
37     RETURN
38

```

CLASSIC F. 1-APR-IV

5.1

04-29-81 09:13

PAGE 24

```
1 SUBROUTINE STATIC(XCUM,YCUM,YCUM2,YCUM3,N1)
2      XCUM=XCUM(1)
3      YCUM=YCUM(1)
4      YCUM2=YCUM(2)
5      YCUM3=YCUM(3)
6      N1=(YCUM2/YCUM-YCUM3/YCUM)*2)
7      C
8      1 FORMAT(1X,1) XCUM*YCUM*YCUM2*YCUM3
9      :*YCUMR=F10.5*X*YEREE,F10.5*EX*VXERR=F10.5*EX,
10      FTUCN
11      FD
```

```
1 SUBROUTINE IDEALST(WAY,K,ECN,DUMS,XSHIFT,YSHIFT)
2   COMPLEX ZD*(N,N),DUMS(N)
3   REAL PI=3.14159
4   PI2=PI*2.0
5   DC=1.1E-8
6   DO 1 J=1,N
7   LU*(1,J)=COMPLEX(0.0,0.0)
8   B/FLCAT(RD)*C*
9   DLS=AT(J)=COMPLEX(0.0,0.0)
10  P1=(J-1)*PI*(J-1-XSHIFT)/FLOAT(N))
11  CONTINUE
12  RETURN
13  END
```

CLASSIC PCHAP, IN L.1 09-29-61 09:13 PAGE 26

```
1      SUBROUTINE CENTROID(X,Y,DATA,N,XCENT,YCENT)
2      CCOMMON DATA(N)
3      DATA=1.0E-6
4      I=1
5      L=1
6      10     X=DATA(I,J)
7      SUMA=0.
8      SUMX=0.
9      SUMY=0.
10     CC=1 J=1,N
11     FOR I=1,DATA(I,J)=LT .0.1*DATA(J,J) DATA(I,J)=0.
12     FOR I=1,DATA(I,J)=LT .0.1*DATA(J,J) DATA(I,J)=0.
13     SUMA=SUMA+REAL(DATA(I,J))
14     SUMX=SUMX+FLOC(I,J)*REAL(DATA(I,J))
15     1     SUMY=SUMY+FLOC(I,J)*REAL(DATA(I,J))
16     XCR=TELE(Y/SUMA*X-0.5
17     YCR=TELE(Y/SUMA*Y-0.5
18     RETURN
19     END
```

1 SPC(LINE) CHACURO(DATA1,0)
2 COPY DATA(N,N),SAVE1,SAVE2
3 12=5/2
4 GO 1 J=1,P2
5 2C 1,J=1,B2
6 SAV1=DATA1,J0
7 DATA1,J0=DATA12,J12,J0
8 DATA12,J12,J0,SAVE1
9 SAVE2=DATA12,J1,J0
10 DATA13,J13,DATA13,J12,J0
11 DATA13,J12,J13,DATA12,J12,J0
12 1 CONJ1,JF
13 RETURN
14 ENO
15 SPC(SI=0
16 SPC(SI=0
17 SPC(MSA,ACC,NOLOC
18 GO TO
19 NEW R1,
20 NEW P4C1T,NCMAP
21 CMAP
22 EDIT MCDFIL,L1
23 EXIT
24 ED BO
25 NEW R1,P0
26 EXEC TOC
FILE LP
REGULAR PIPELINE
VERIFY
STOP
OVERPLAY MCDFIL
CATALOG MCDFIL
EXIT
SAVE CI,C
263 PASS USLUSL,SI=USL
SECD FORTRA
SREG RLS ALL

Plotting Commands

```

LS GS,.SCA*
FOR .(6E12 S)
CAL LL,1,-.367 UR,20.,.148
AXES INTERSECT,1.,-.367
XL 1,
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED X MINUS POSITION MEAN ERROR +/- SIGMA"
LA S,.1 POS .(COU,NZ,ALP,NF,UAB,SD)
LA S,.1 ADD
EN M,2 NP 20
PLOT S,6 SS,.05
EN M,2 NP 20
PLOTS S,114 SS,.05
EN M,2 NP 20
PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RECAL LL,1,-.328 UR,20.,.125
AXES INTERSECT,1.,-.328
XL 1,
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED Y MINUS POSITION MEAN ERROR +/- SIGMA"
LA S,.1 POS .(COU,NZ,ALP,NF,UAB,SD)
LA S,.1 ADD
EN M,2 NP 20
PLOT S,6 SS,.05
EN M,2 NP 20
PLOT S,114 SS,.05
EN M,2 NP 20
PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RECAL LL,1,-.367 UR,20.,.148
AXES INTERSECT,1.,-.367
XL 1,
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED Y MINUS POSITION MEAN ERROR +/- SIGMA"
LA S,.1 POS .(COU,NZ,ALP,NF,UAB,SD)
LA S,.1 ADD
EN M,2 NP 40
PLOT S,6 SS,.05
EN M,2 NP 40
PLOT S,114 SS,.05
EN M,2 NP 40
PLOT S,112 SS,.05
85
FRAME PAGE
GRA 1
RECAL LL,1,-.367 UR,20.,.148
AXES INTERSECT,1.,-.367
XL 1,
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED X PLUS POSITION MEAN ERROR +/- SIGMA"
LA S,.1 POS .(COU,NZ,ALP,NF,UAB,SD)
LA S,.1 ADD
EN M,2 NP 20
PLOT S,6 SS,.05
EN M,2 NP 20
PLOT S,114 SS,.05
EN M,2 NP 20
PLOT S,112 SS,.05
43
FRAME PAGE
GRA 1
RECAL LL,1,-.367 UR,20.,.148
AXES INTERSECT,1.,-.367
XL 1,
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED X PLUS POSITION MEAN ERROR +/- SIGMA"
LA S,.1 POS .(COU,NZ,ALP,NF,UAB,SD)
LA S,.1 ADD
EN M,2 NP 20
PLOT S,6 SS,.05
EN M,2 NP 20
PLOT S,114 SS,.05
EN M,2 NP 20
PLOT S,112 SS,.05

```

```

: 'NAME PAGE
GRA 1
RECAL LL,1,-.328 UR,20.,.125
AXES INTERSECT,1.,-.328
XL 1,
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED Y PLUS POSITION MEAN ERROR +/- SIGMA"
LA S,.1 POS .(COU,NZ,ALP,NF,UAB,SD)
LA S,.1 ADD
EN M,2 NP 20
PLOT S,6 SS,.05
64
EN M,2 NP 20
PLOT S,5,6 SS,.05
EN M,2 NP 20
PLOT S,114 SS,.05
EN M,2 NP 20
PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RECAL LL,1,-.367 UR,20.,.148
AXES INTERSECT,1.,-.367
XL 1,
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED X POSITION MEAN ERROR +/- SIGMA"
LA S,.1 POS .(COU,NZ,ALP,NF,UAB,SD)
LA S,.1 ADD
EN M,2 NP 40
PLOT S,6 SS,.05
EN M,2 NP 40
PLOT S,114 SS,.05
EN M,2 NP 40
PLOT S,112 SS,.05
85
FRAME PAGE
GRA 1
RECAL LL,1,-.367 UR,20.,.148
AXES INTERSECT,1.,-.367
XL 1,
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED X POSITION MEAN ERROR +/- SIGMA"
LA S,.1 POS .(COU,NZ,ALP,NF,UAB,SD)
LA S,.1 ADD
EN M,2 NP 40
PLOT S,6 SS,.05
EN M,2 NP 40
PLOT S,114 SS,.05
EN M,2 NP 40
PLOT S,112 SS,.05

```

LINE
85

```
L PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RECAL LL,1,-.328 UR,20...125
AXES INTERSECT,1.,-.328
XL 1,1
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED Y POSITION MEAN ERROR +/- SIGMA"
LA S,.1 POS "(COU,NZ,ALP,NF,UAB,SD),"
LA S,.1 ADD
LA S,.1 ADD
EN R,2 NP,40
PLOT S,6 SS,.05
EN R,2 NP,40
PLOT S,114 SS,.05
EN R,2 NP,40
PLOT S,112 SS,.05
PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RECAL LL,1,-.390 UR,20...072
AXES INTERSECT,1.,-.390
106
L AXES INTERSECT,1.,-.390
XL 1,1
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED X MINUS POSITION CENTROID MEAN ERROR +/- SIGMA"
LA S,.1 POS "(COU,NZ,ALP,NF,UAB,SD),"
LA S,.1 ADD
LA S,.1 ADD
EN R,2 NP,20
PLOT S,6 SS,.05
EN R,2 NP,20
PLOT S,114 SS,.05
EN R,2 NP,20
PLOT S,112 SS,.05
PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RECAL LL,1,-.318 UR,20...08
AXES INTERSECT,1.,-.318
XL 1,1
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED Y MINUS POSITION CENTROID MEAN ERROR +/- SIGMA"
LA S,.1 POS "(COU,NZ,ALP,NF,UAB,SD),"
LA S,.1 ADD
LA S,.1 ADD
EN R,2 NP,20
PLOT S,6 SS,.05
EN R,2 NP,20
PLOT S,114 SS,.05
EN R,2 NP,20
PLOT S,112 SS,.05
PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RECAL LL,1,-.318 UR,20...08
AXES INTERSECT,1.,-.318
XL 1,1
YL 1,.05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED Y MINUS POSITION CENTROID MEAN ERROR +/- SIGMA"
LA S,.1 POS "(COU,NZ,ALP,NF,UAB,SD),"
LA S,.1 ADD
LA S,.1 ADD
EN R,2 NP,20
PLOT S,6 SS,.05
EN R,2 NP,20
```

LINKE

169

```
N R,2 NP,20
PLOT S,114 SS,.05
EN R,2 NP,20
PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RECAL LL:1:-390 UR,20...072
AXES INTERSECT,1.,-.390
XL 1,1
YL 1,-05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED X POSITION CENTROID MEAN ERROR +/- SIGMA"
LA S,.1 POS "(COU,NZ,ALP,NF,UAB,SD)".
LA S,.1 ADD
EN R,2 NP,40
PLOT S,6 SS,.05
EN R,2 NP,40
PLOT S,114 SS,.05
EN R,2 NP,40
PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RECAL LL,1,.-.318 UR,20...08
190
RECAL LL,1,.-.318 UR,20...08
AXES INTERSECT,1.,-.318
XL 1,1
YL 1,-05 D,3
XT "FRAME"
YT "ERROR (PIXELS)"
TI "ESTIMATED Y POSITION CENTROID MEAN ERROR +/- SIGMA"
LA S,.1 POS "(COU,NZ,ALP,NF,UAB,SD)".
LA S,.1 ADD
EN R,2 NP,40
PLOT S,5,6 SS,.05
EN R,2 NP,40
PLOT S,114 SS,.05
EN R,2 NP,40
PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RECAL LL,1,.-2.1 UR,20...1.9
AXES INTERSECT,1.,-2.1
XL 1,1
YL 1,-4 D,3
XT "FRAME"
211
RECAL LL,1,.-2.1 UR,20...1.9
AXES INTERSECT,1.,-2.1
XL 1,1
YL 1,-4 D,3
XT "FRAME"
YT "ERROR (INTENSITY)"
TI "MEAN ERROR +/- SIGMA OF ESTIMATED INTENSITY PROFILE"
LA S,.1 POS "(COU,NZ,ALP,NF,UAB,SD)".
LA S,.1 ADD
EN R,2 NP,20
PLOT S,6 SS,.05
```

LINE 253
EN R, 2 NP,.20
PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RECAL LL,1,-1.80 UR,20,-3.15
AXES INTERSECT,1,-1.8
XL 1.
YL 1.
XT "FRAME"
YT "ERROR (INTENSITY)"
TI "MEAN ERROR +/- SIGMA OF ESTIMATED DM/DY"
L S,.1 POS "(CQU,MZ,ALP,NF,UAB,SD)"
LA S,.1 ADD
EN R,2 NP,.20
PLOT S,6 SS,.05
EN R,2 NP,.20
PLOT S,114 SS,.05
EN R,2 NP,.20
PLOT S,112 SS,.05
FRAME PAGE
GRA 1
RETN
258
REM TIPCOM
CAT TIPCOM

AD-A115 581

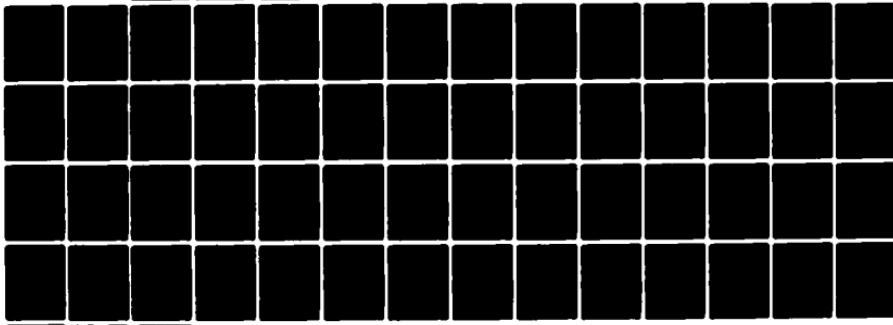
AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHO--ETC F/G 12/1
ENHANCED TRACKING OF AIRBORNE TARGETS USING FORWARD LOOKING INF--ETC(U)
DEC 81 S K ROGERS

UNCLASSIFIED

AFIT/GEO/EE/B1D-5

NL

4 of 4
NOV 81



END
DATE
FILED
7-82
DTIC

Appendix I
Plots of Tracking Errors

The sequence of these plots is explained in Chapter VI. This appendix contains plots for four settings of parameters which control intensity function derivation and Kalman filter characteristics. The legend on each plot provides information on the target's spread parameter, the number of zeros which are padded, the smoothing constant, the number of frequencies to zero, the background noise strength, the standard deviation of the discrete time noise driving the target dynamic states of the filter respectively.

Pad Zeros

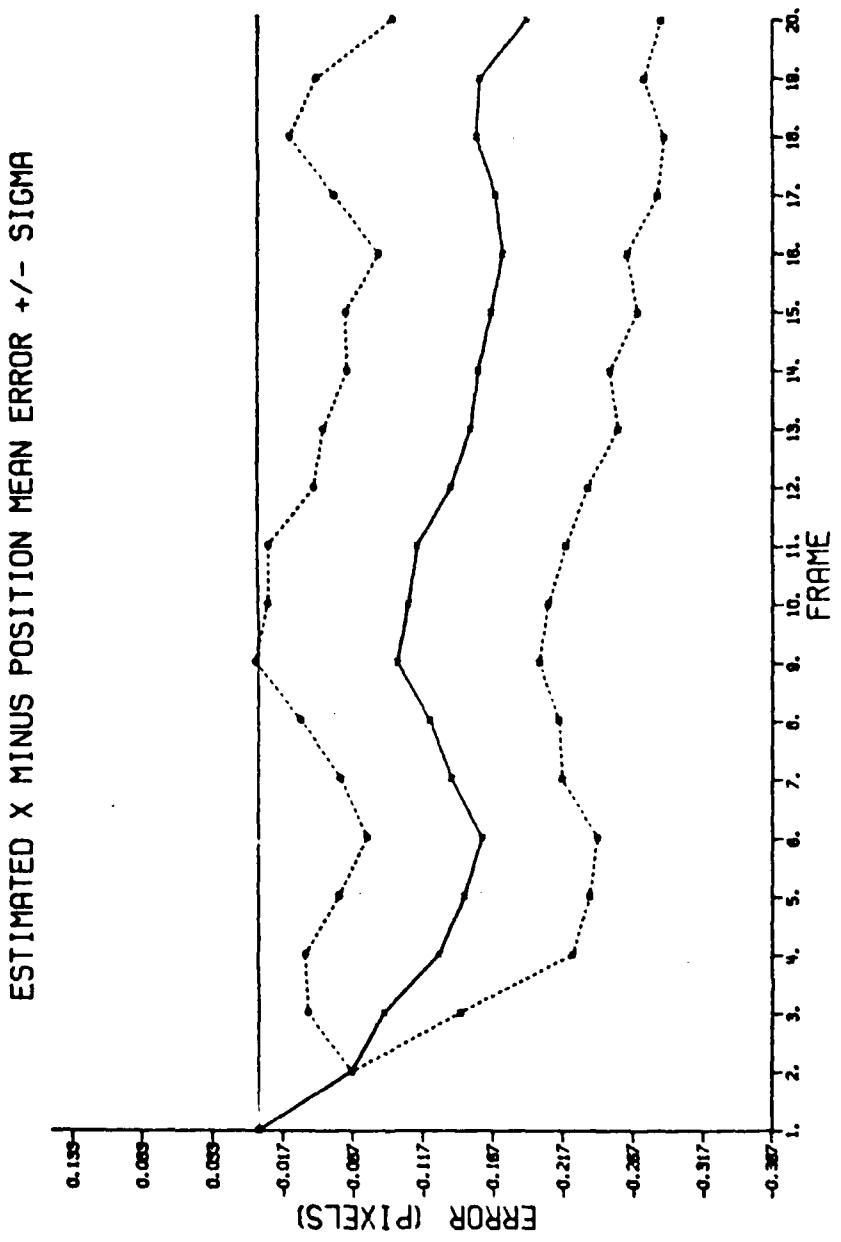


Figure 1-1. Pad Zeros & Minus Errors

ESTIMATED Y MINUS POSITION MEAN ERROR +/- SIGMA

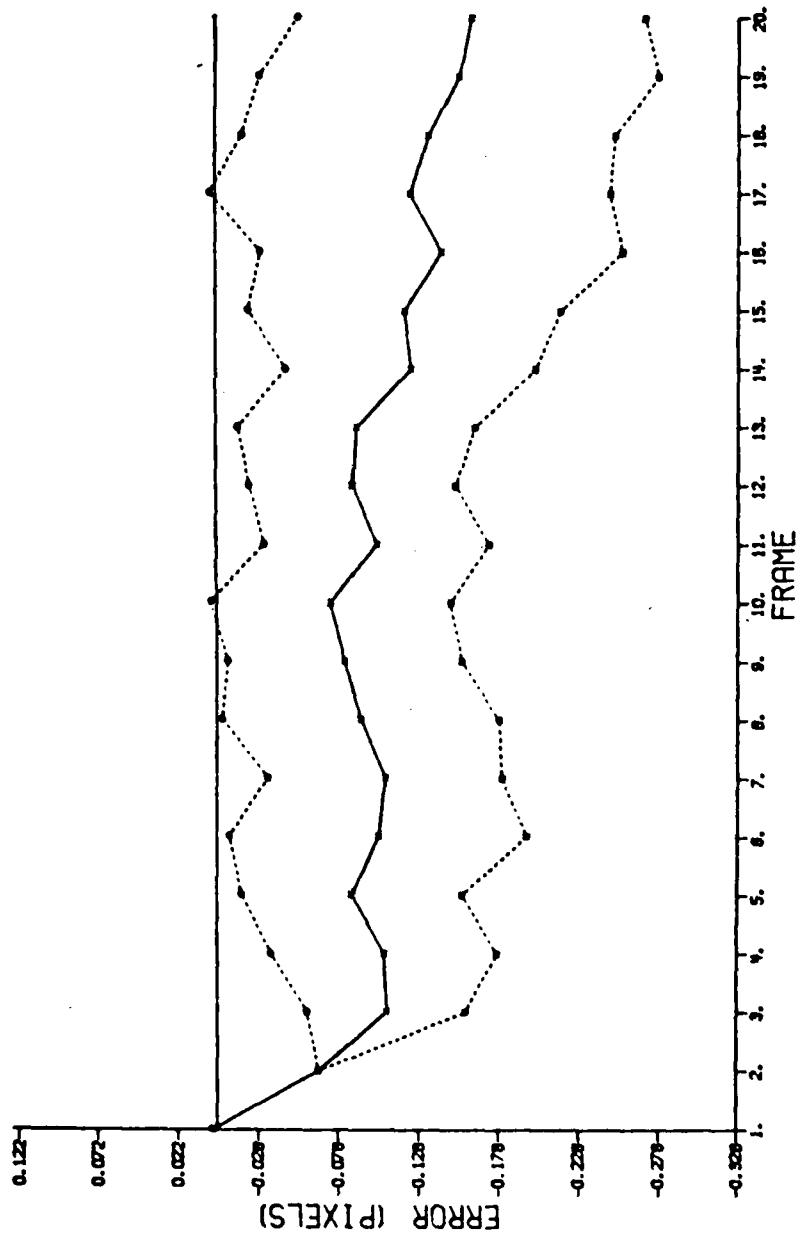


Figure 1-2. Pad Zeros Y Minus Errors

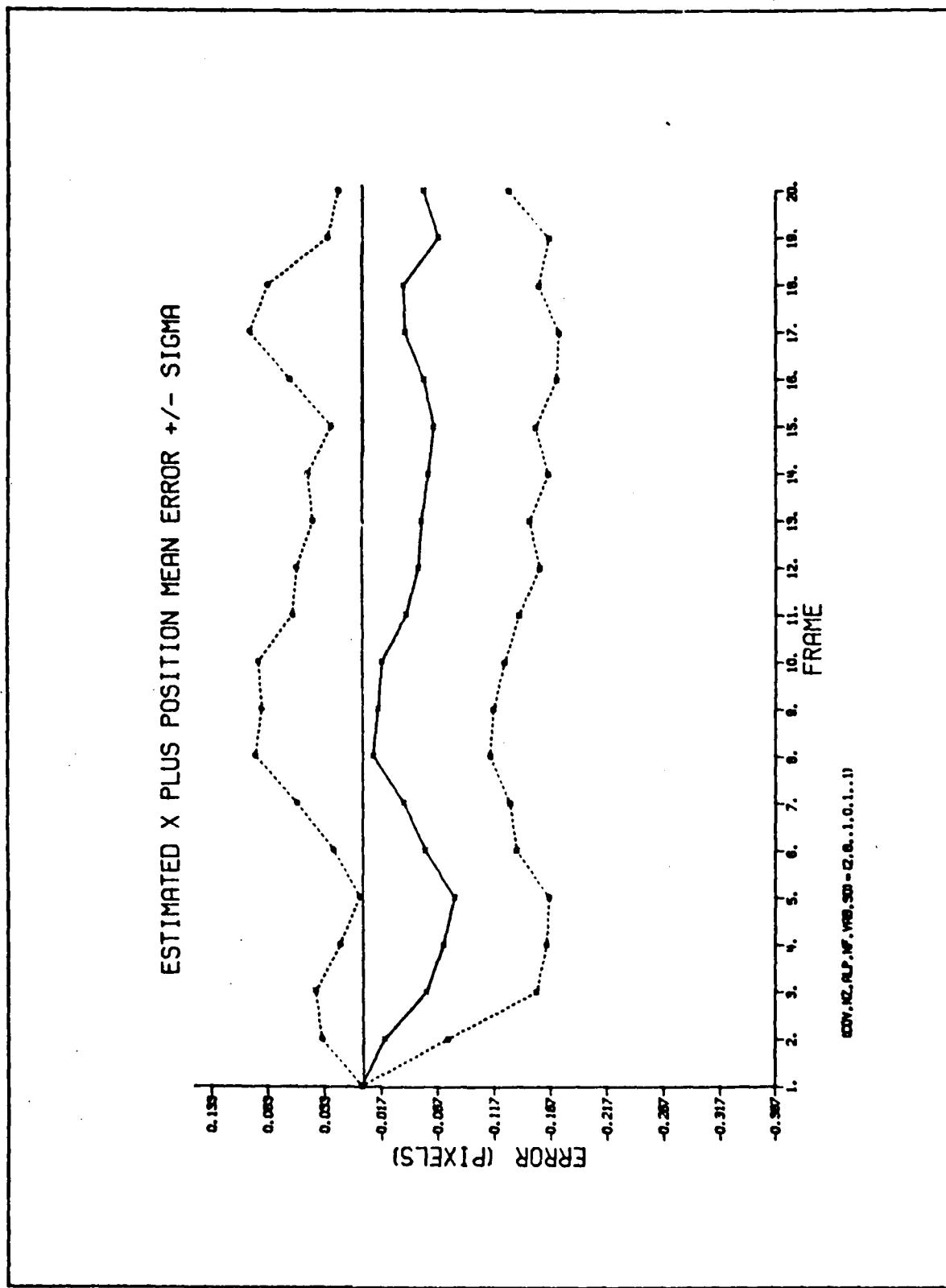


Figure 1-3. Pad Zeros X Plus Errors

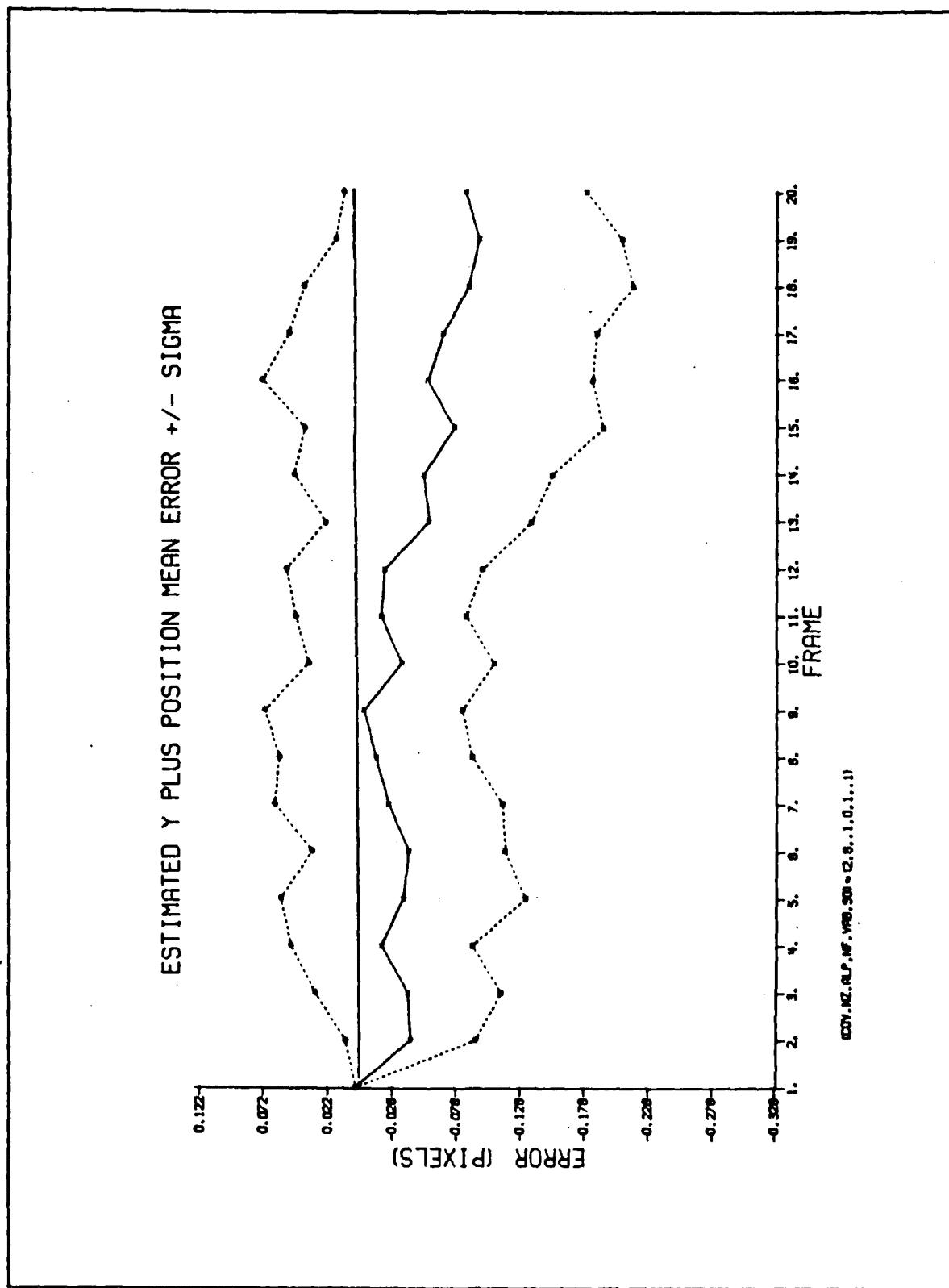


Figure 1-4. Pad Zeros Y Plus Errors

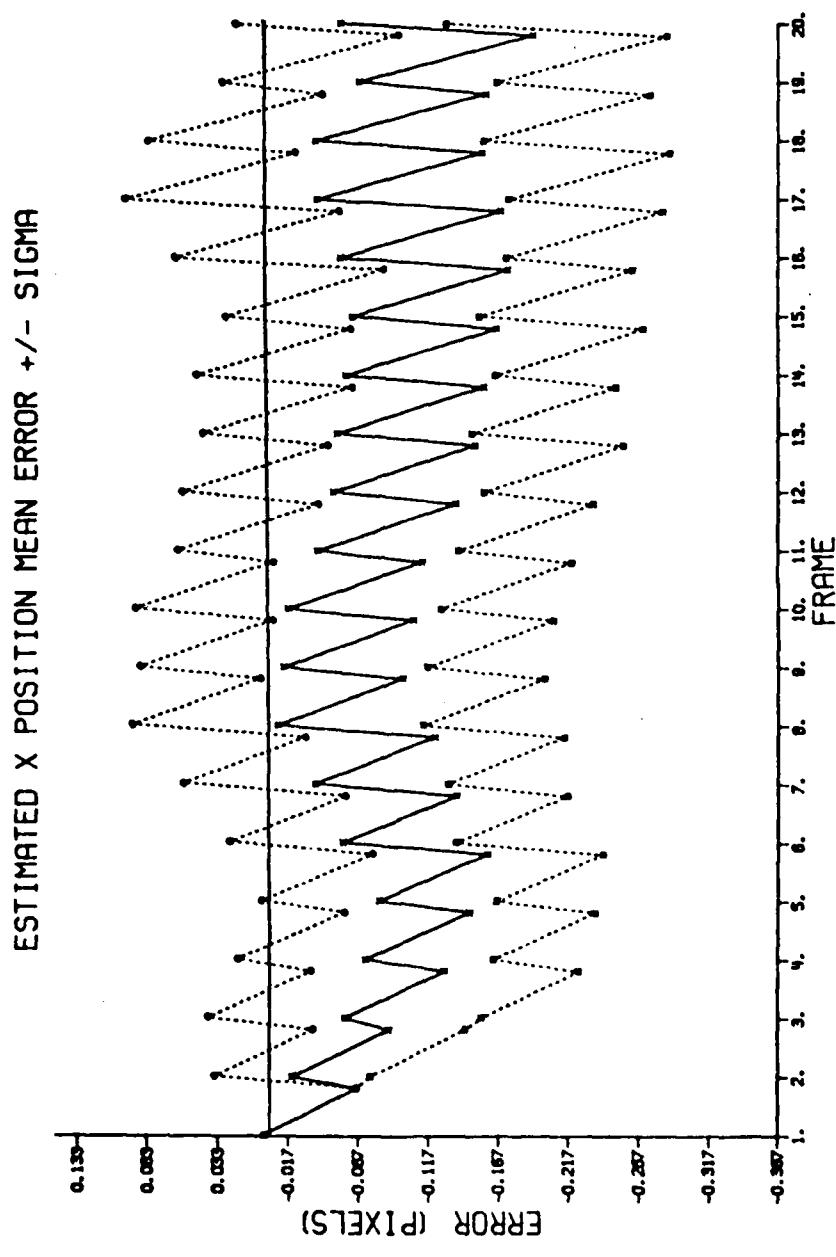


Figure 1-5. Pad Zeros X Position Errors

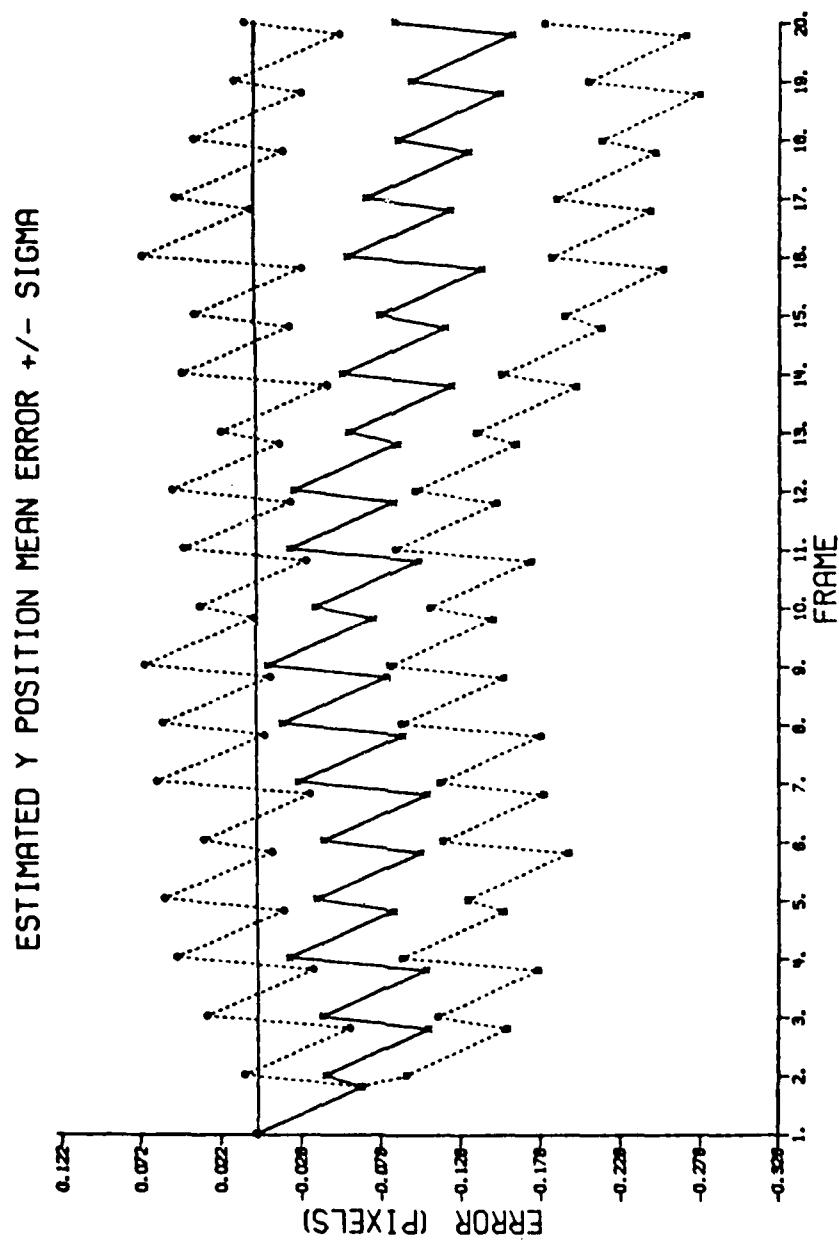


Figure 1-6. Pad Zeros Y Position Errors

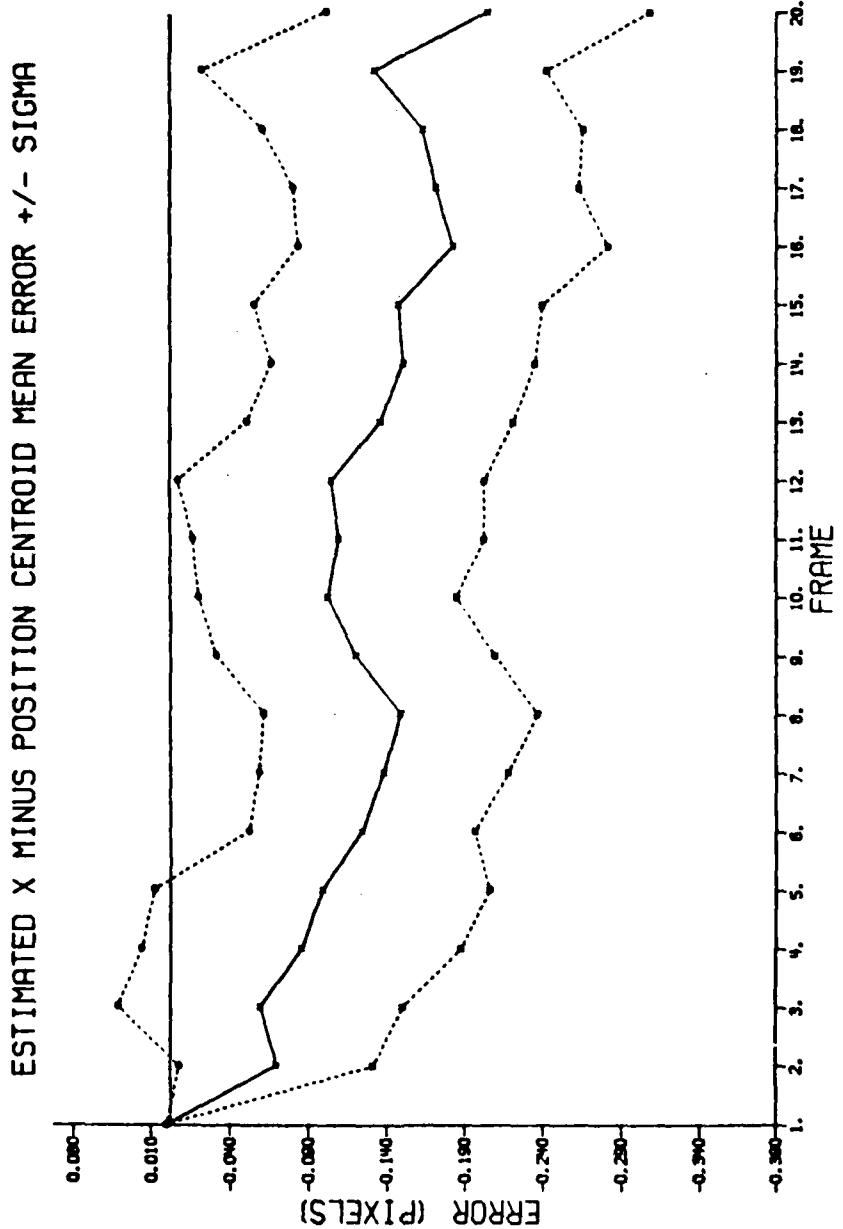


Figure 1-7. Pad Zeros X Centroid Minus Errors

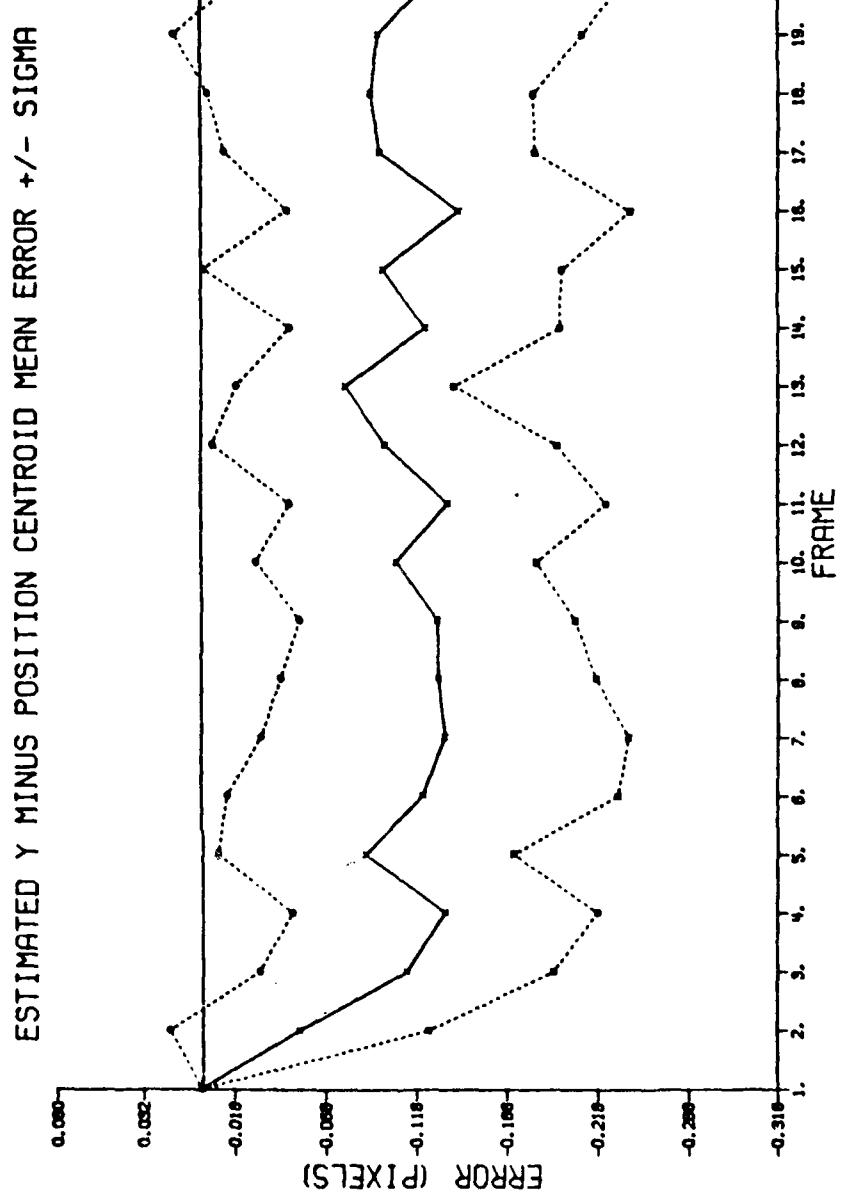


Figure 1-8. Pad Zeros Y Centroid Minus Errors

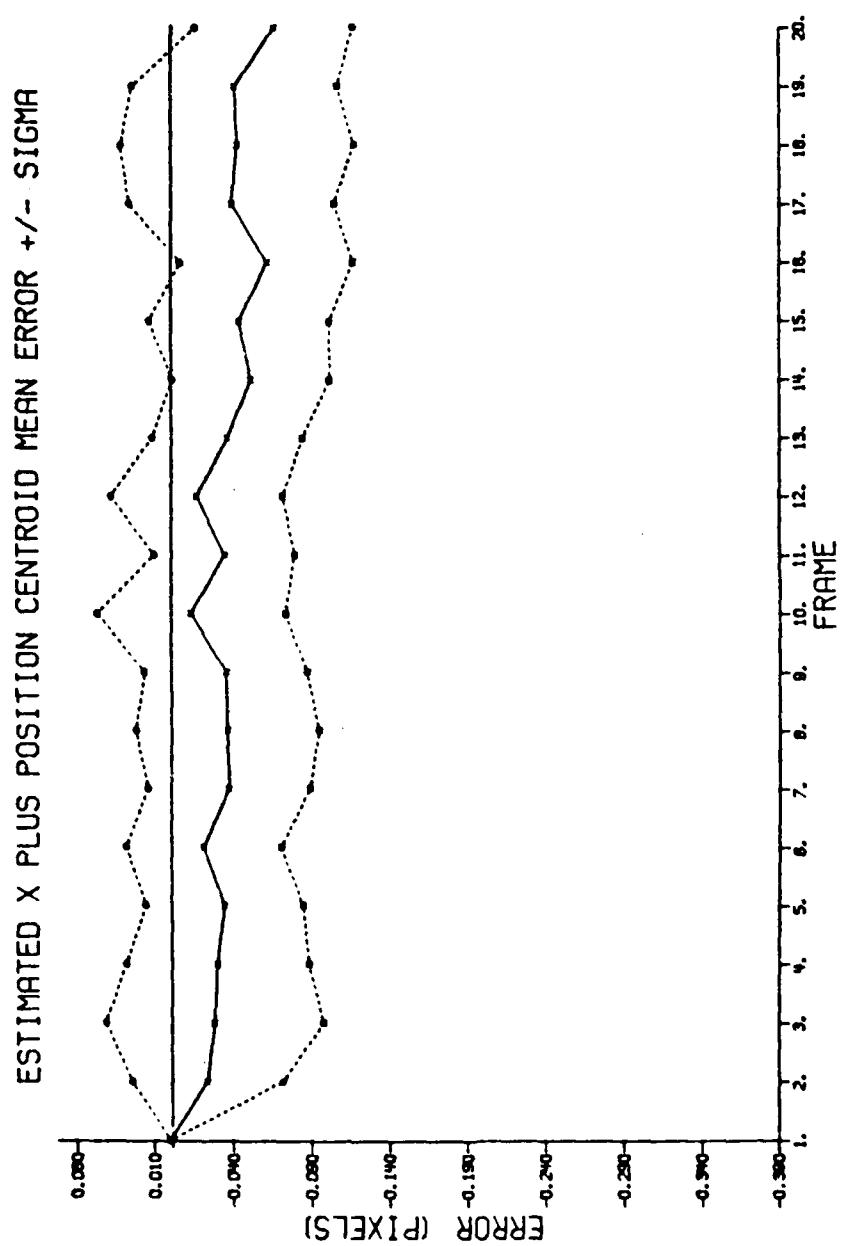


Figure 1-9. Pad Zeros X Centroid Plus Errors

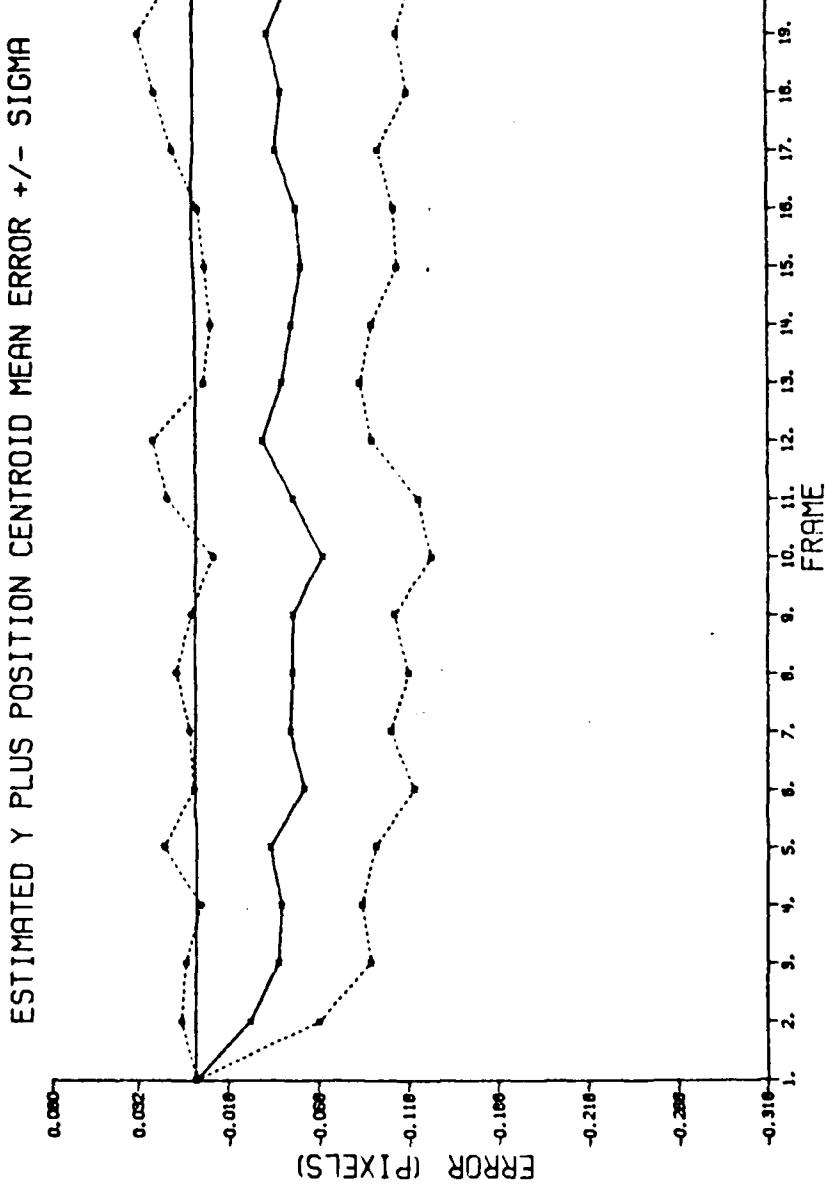


Figure 1-10. Pad Zeros Y Centroid Plus Errors

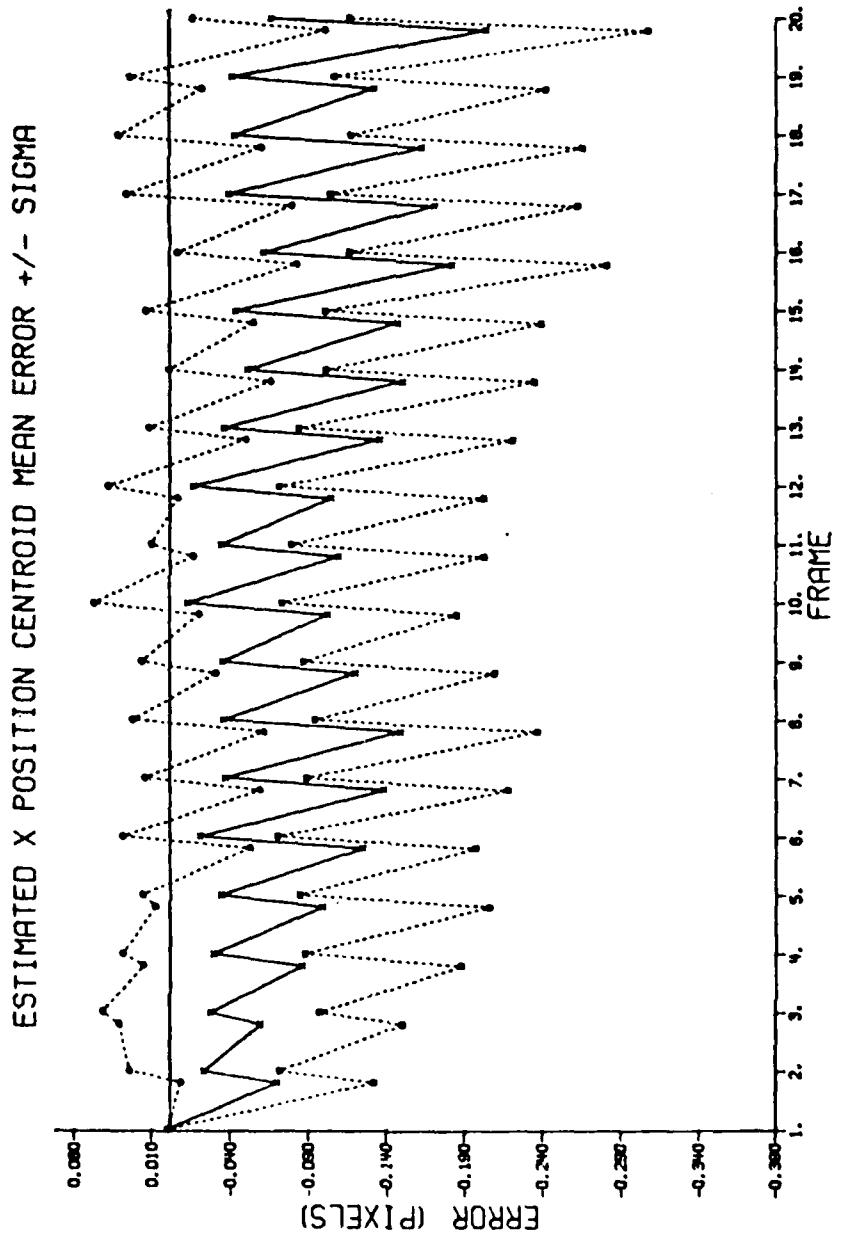


Figure 1-11. Pad Zeros X Centroid Position Errors

ESTIMATED Y POSITION CENTROID MEAN ERROR +/- SIGMA

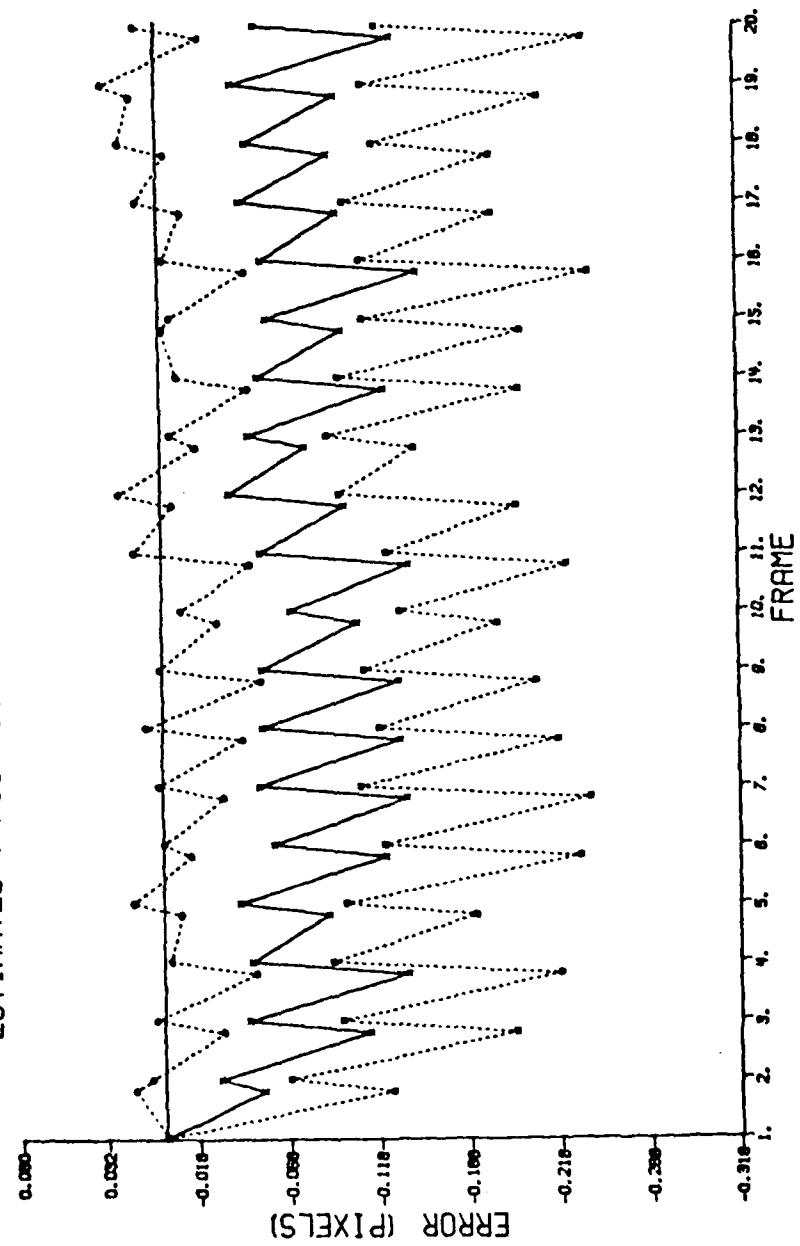


Figure 1-12. Pad Zeros Y Centroid Position Errors

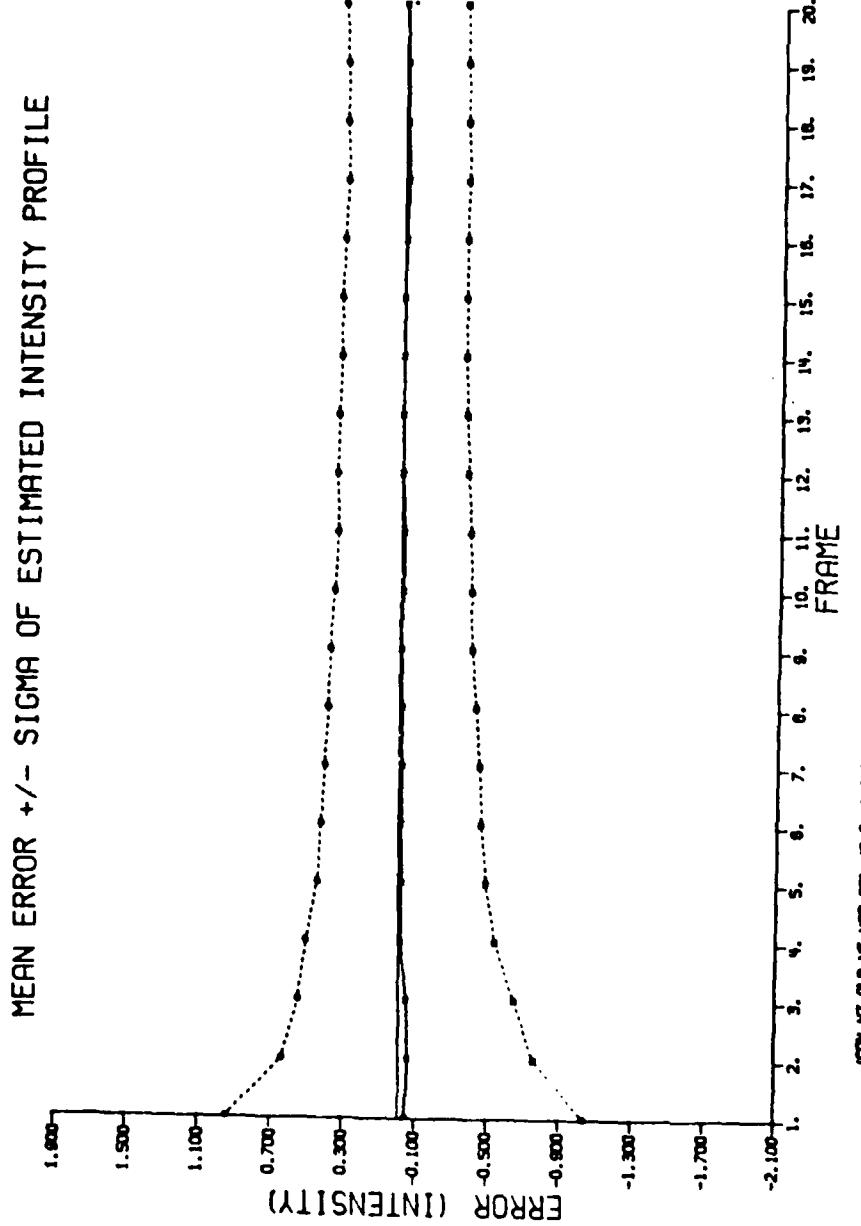


Figure 1-13. Pad Zeros Error of Estimated \hat{h}

MEAN ERROR +/- SIGMA OF ESTIMATED DH/DX

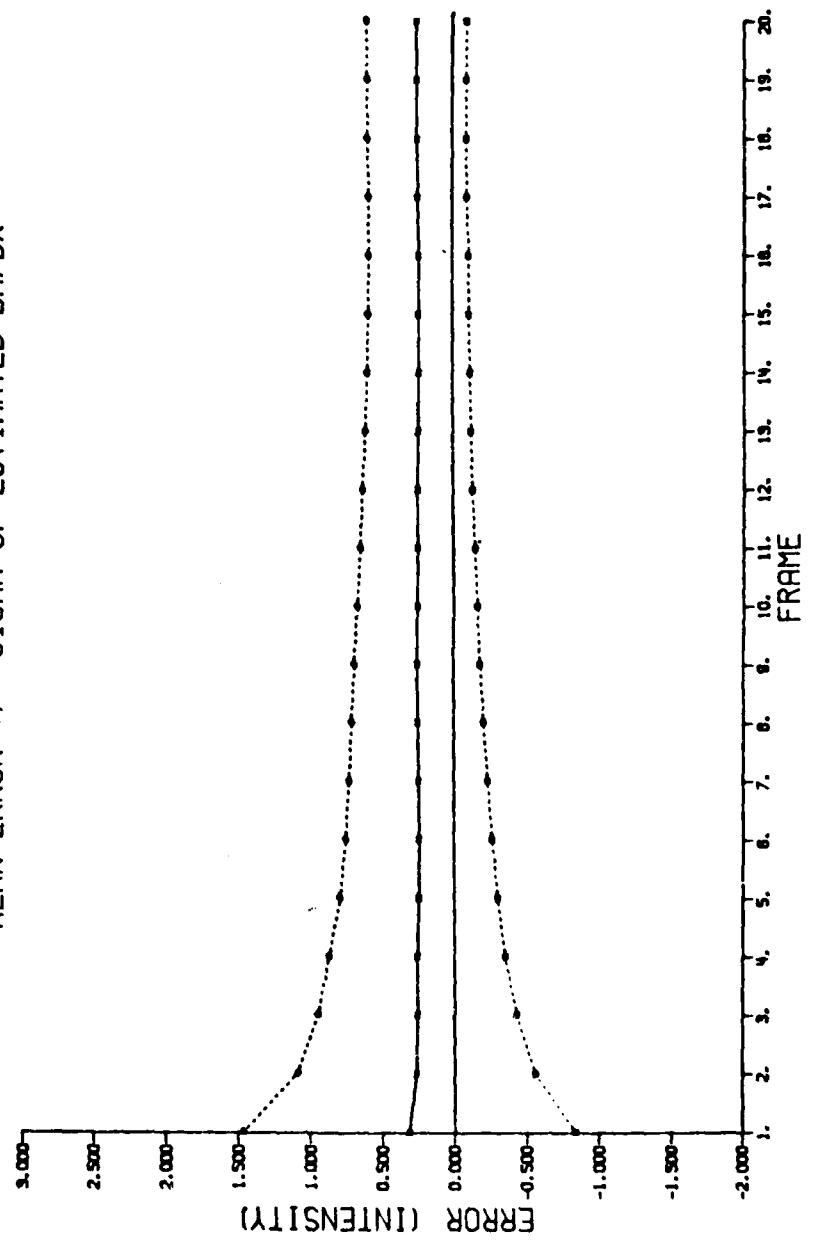


Figure 1-14. Pad Zeros Error of Estimated Dh/Dx

MEAN ERROR +/- SIGMA OF ESTIMATED DH/DY

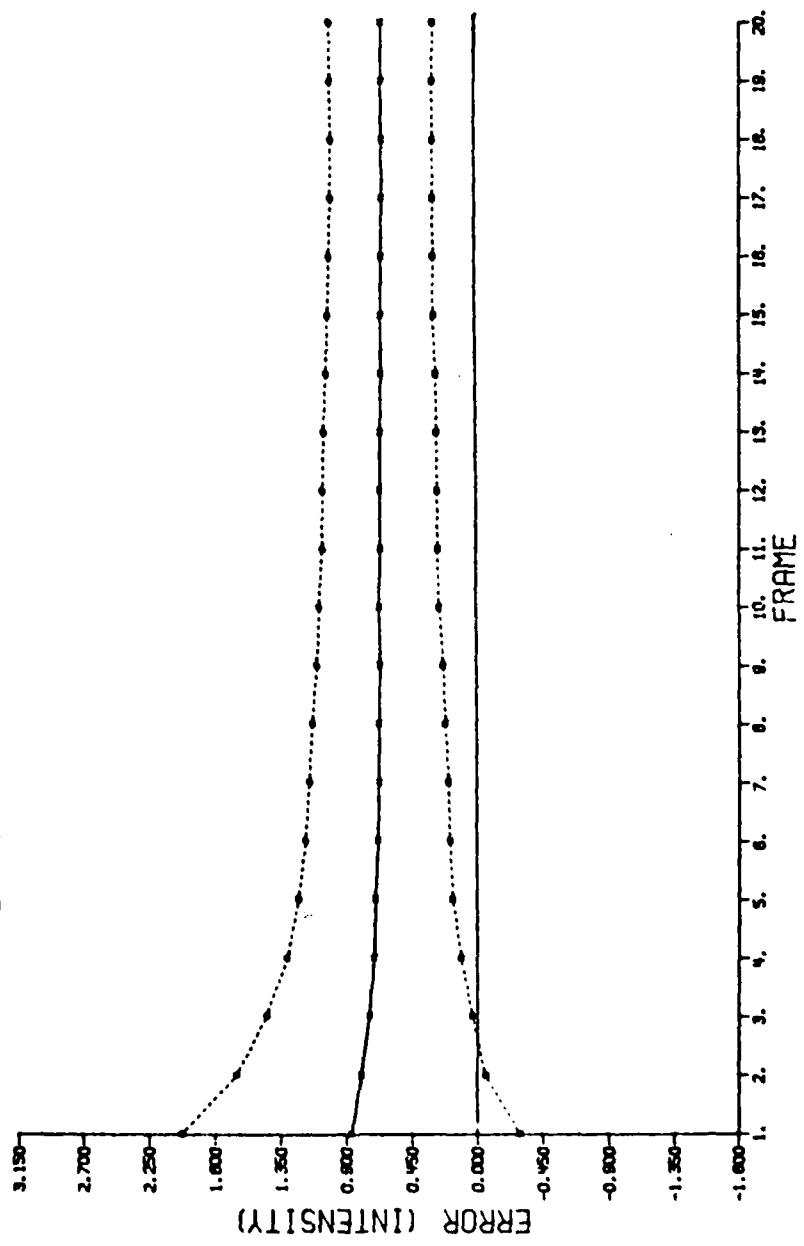


Figure 1-15. Pad Zeros Error of Estimated Dh/Dy

Removing Two Highest
Spatial Frequencies

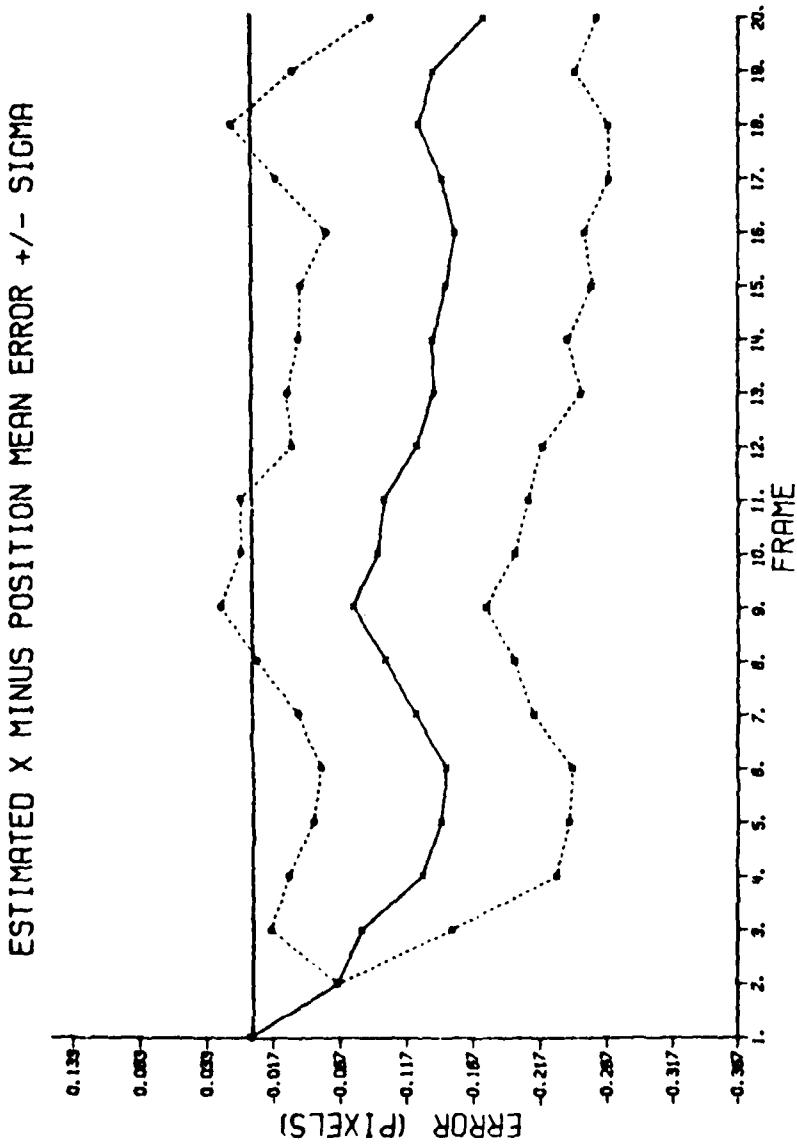


Figure 1-16. Removing Two Highest Spatial Frequencies
X Minus Errors

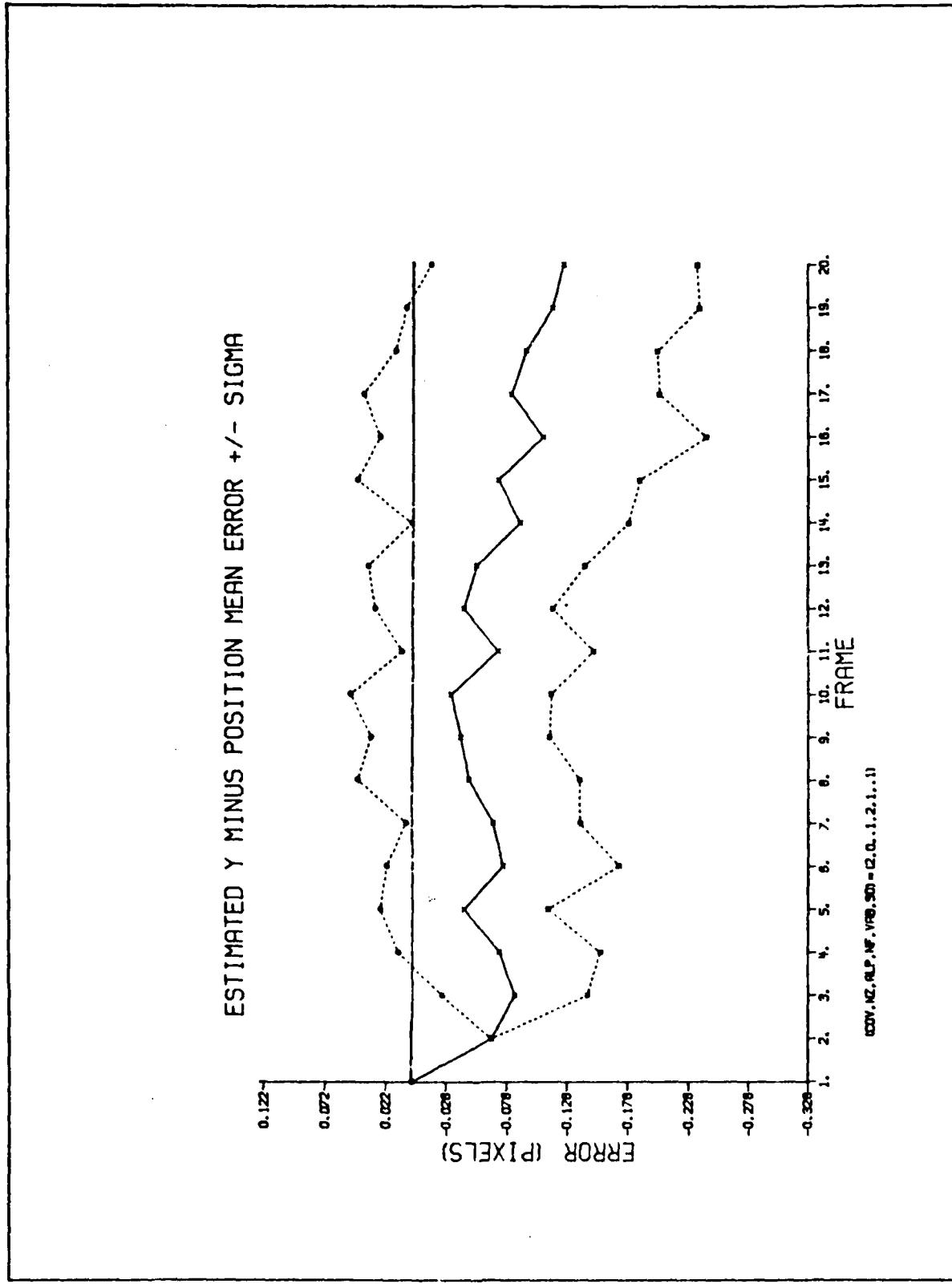


Figure 1-17. Removing Two Highest Spatial Frequencies
Y Minus Errors

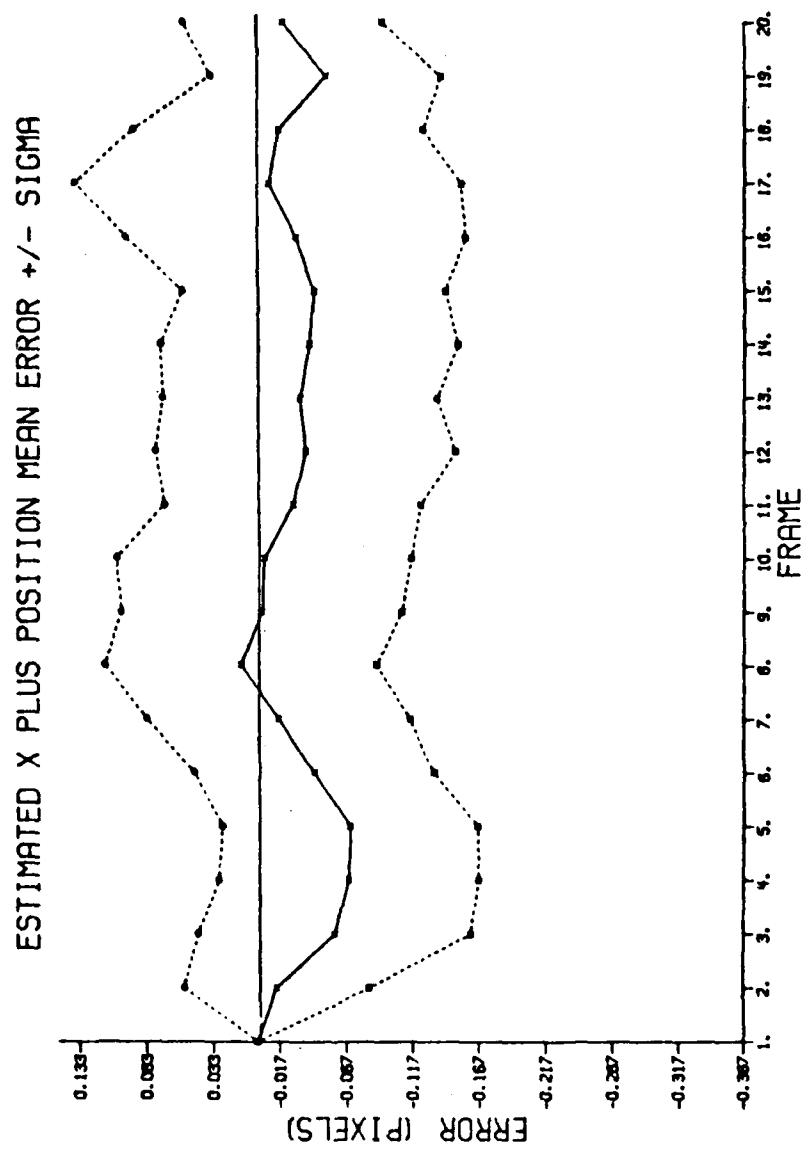


Figure 1-18. Removing Two Highest Spatial Frequencies X Plus Errors

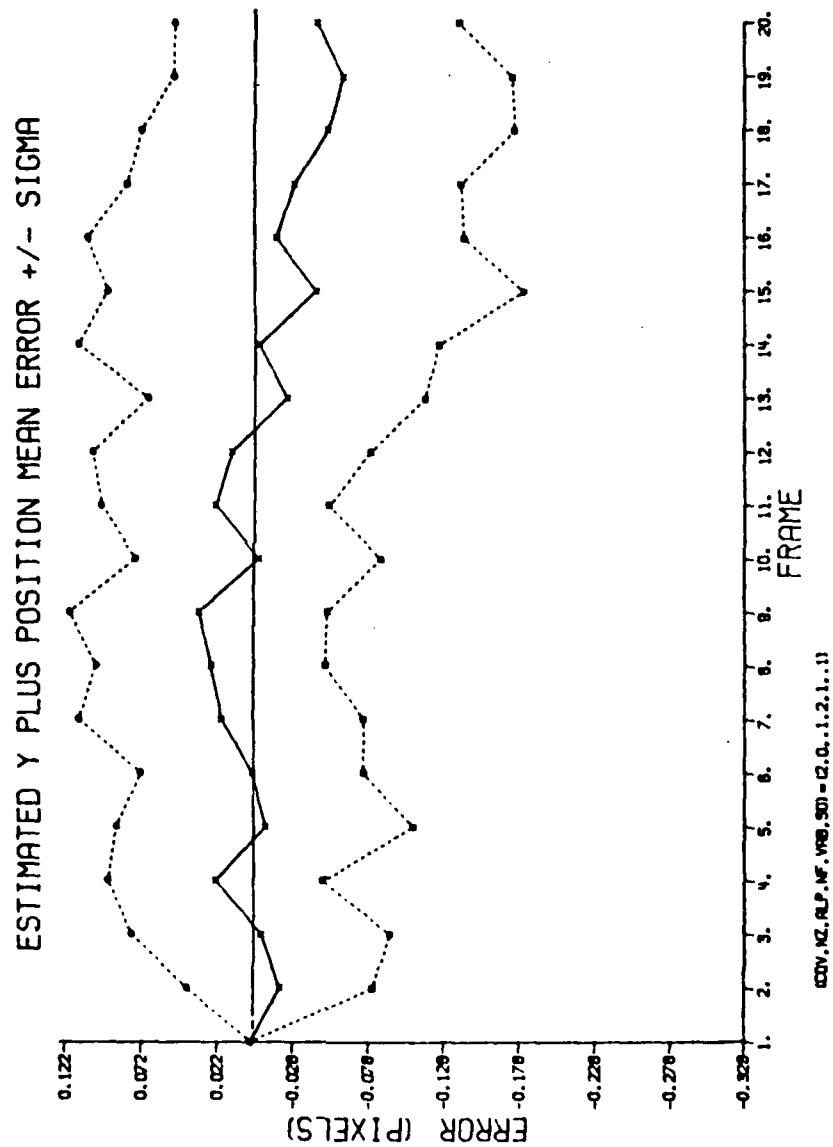


Figure 1-19. Removing Two Highest Spatial Frequencies Y Plus Errors

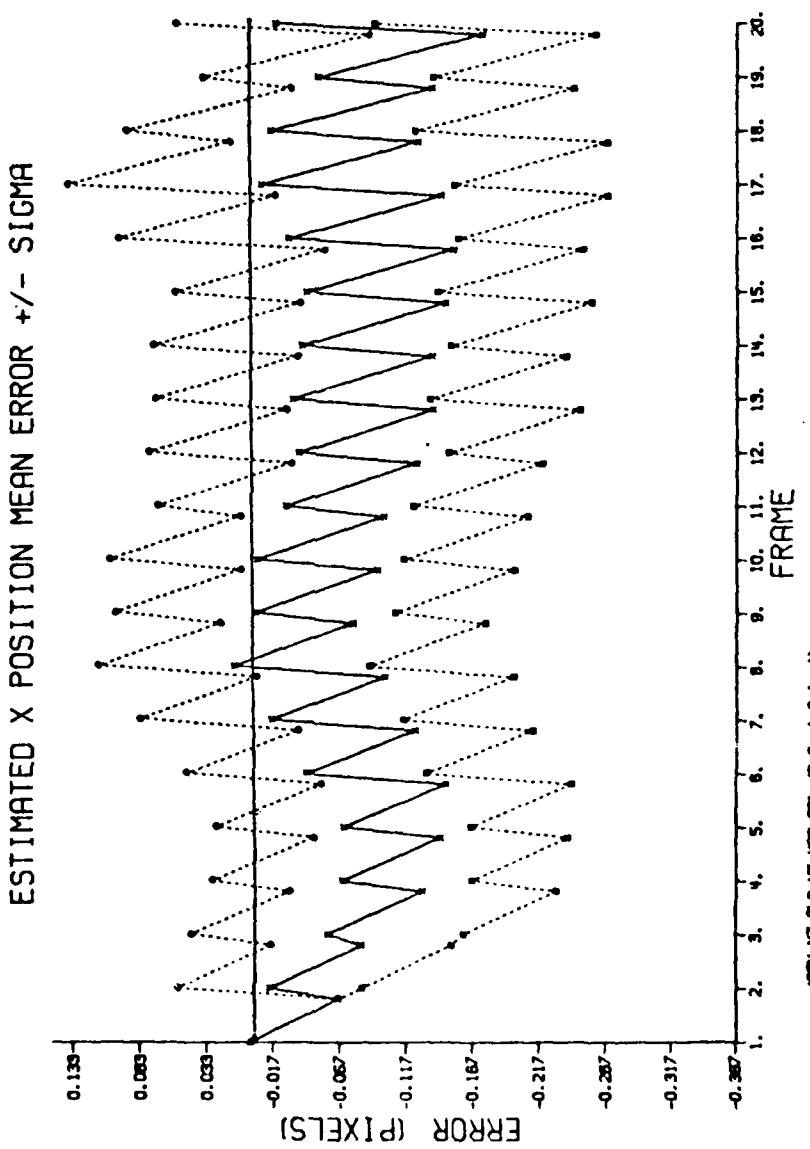


Figure 1-20. Removing Two Highest Spatial Frequencies
X Position Errors

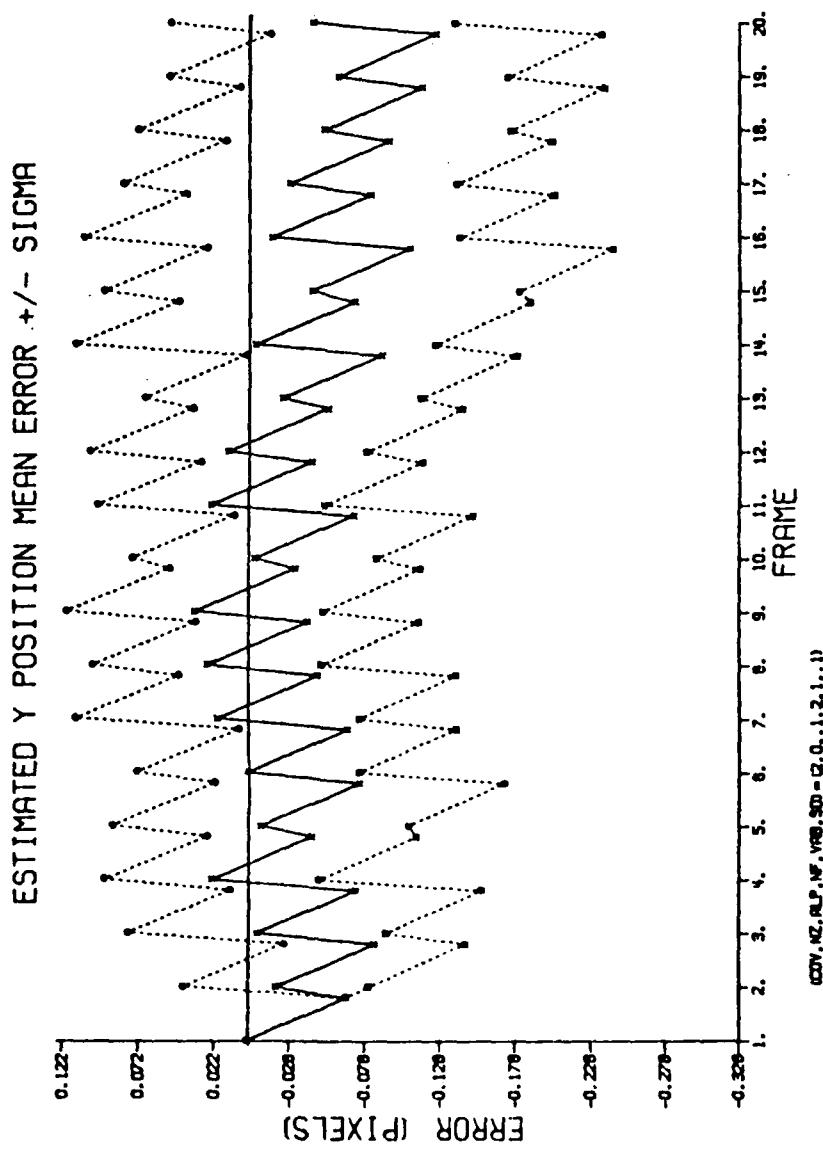


Figure 1-21. Removing Two Highest Spatial Frequencies Y Position Errors

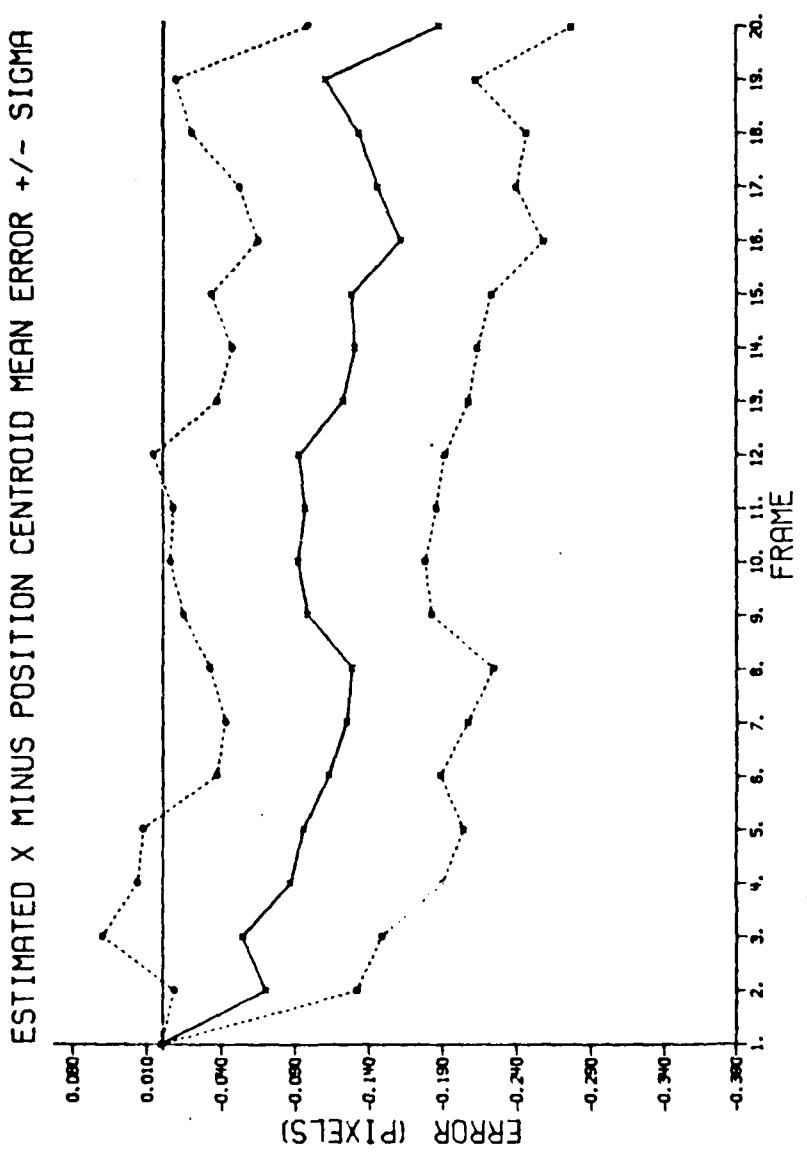


Figure 1-22. Removing Two Highest Spatial Frequencies
X Centroid Minus Errors

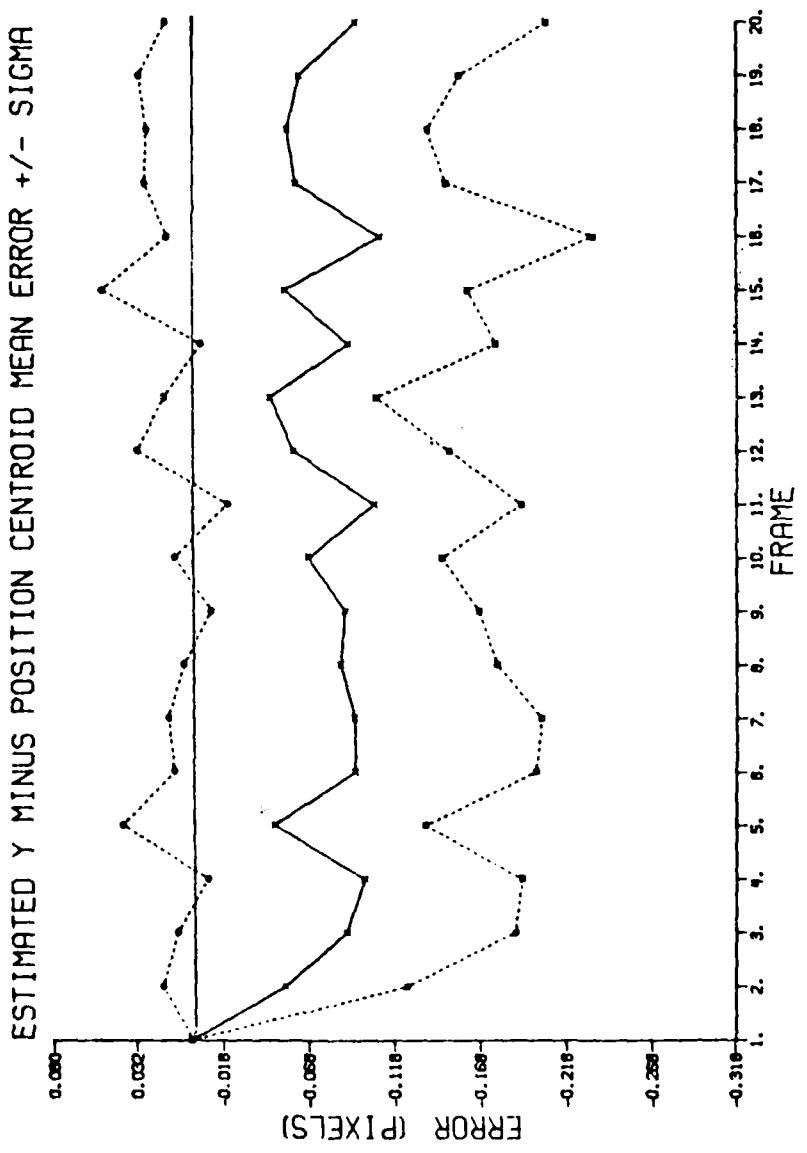


Figure 1-23. Removing Two Highest Spatial Frequencies
Y Centroid Minus Errors

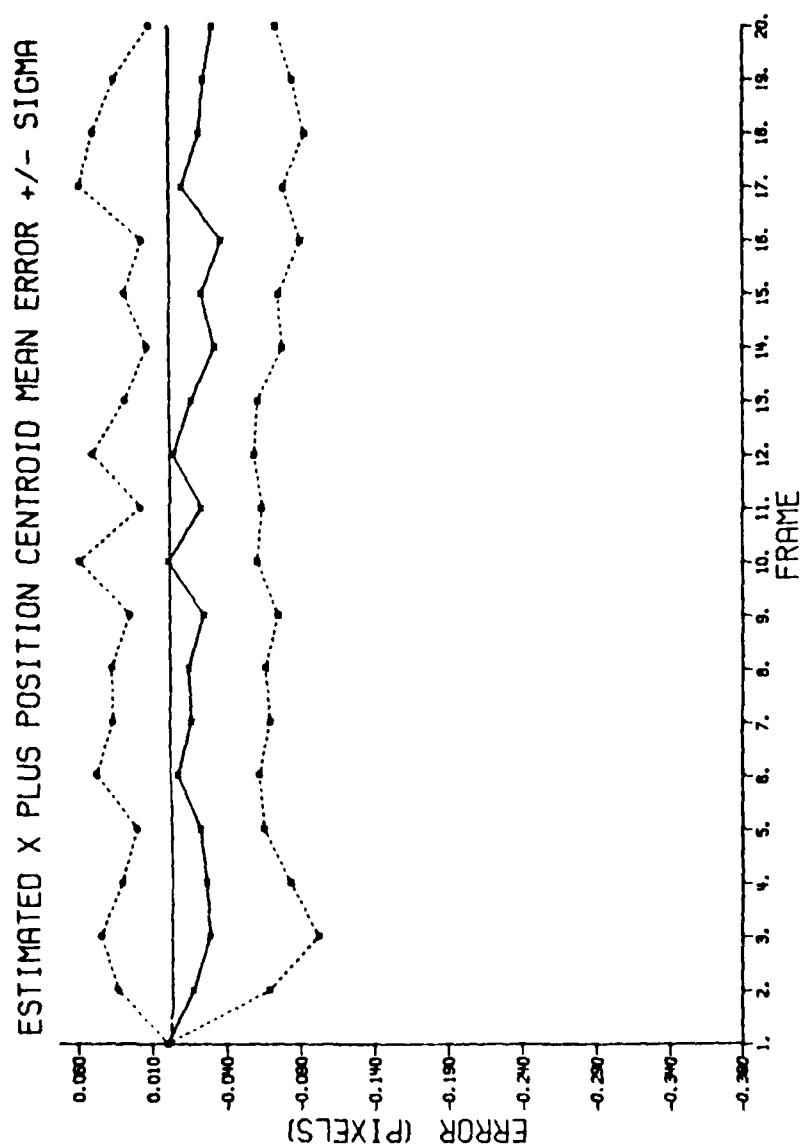


Figure 1-24. Removing Two Highest Spatial Frequencies
X Centroid Plus Errors

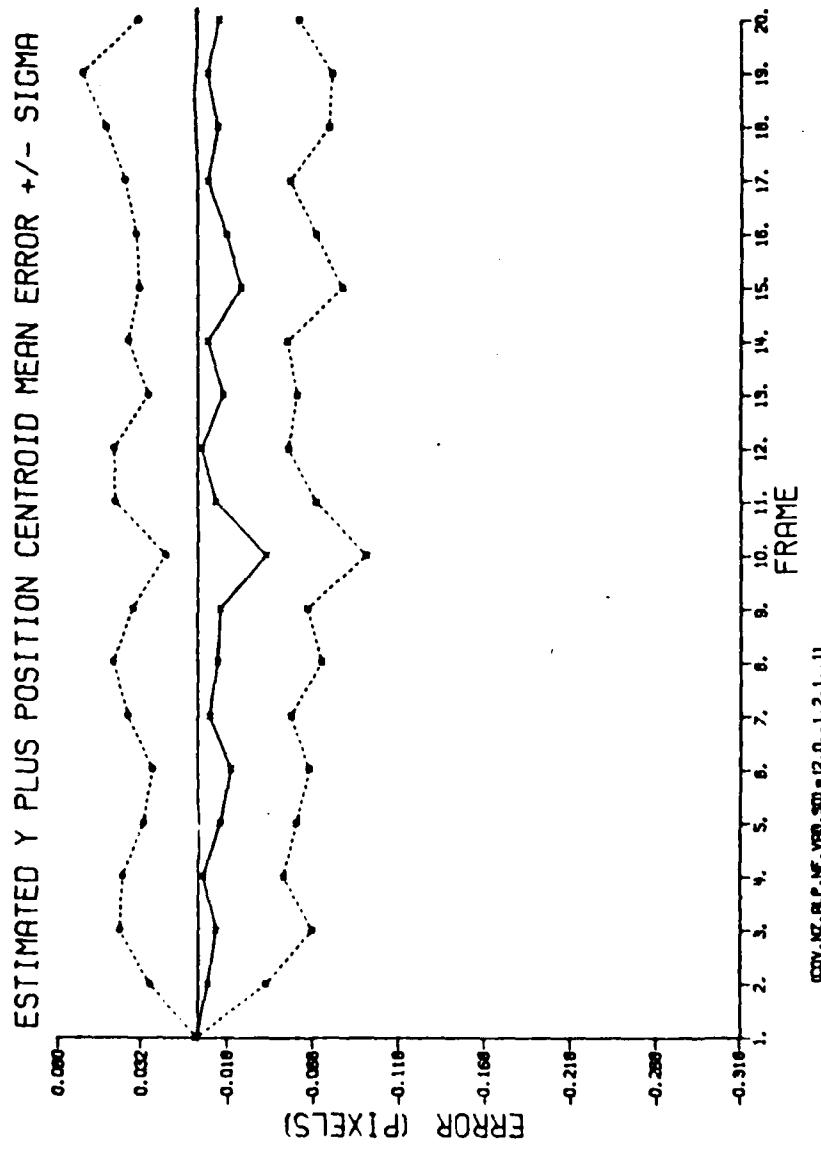


Figure 1-25. Removing Two Highest Spatial Frequencies
Y-Centroid Plus Errors

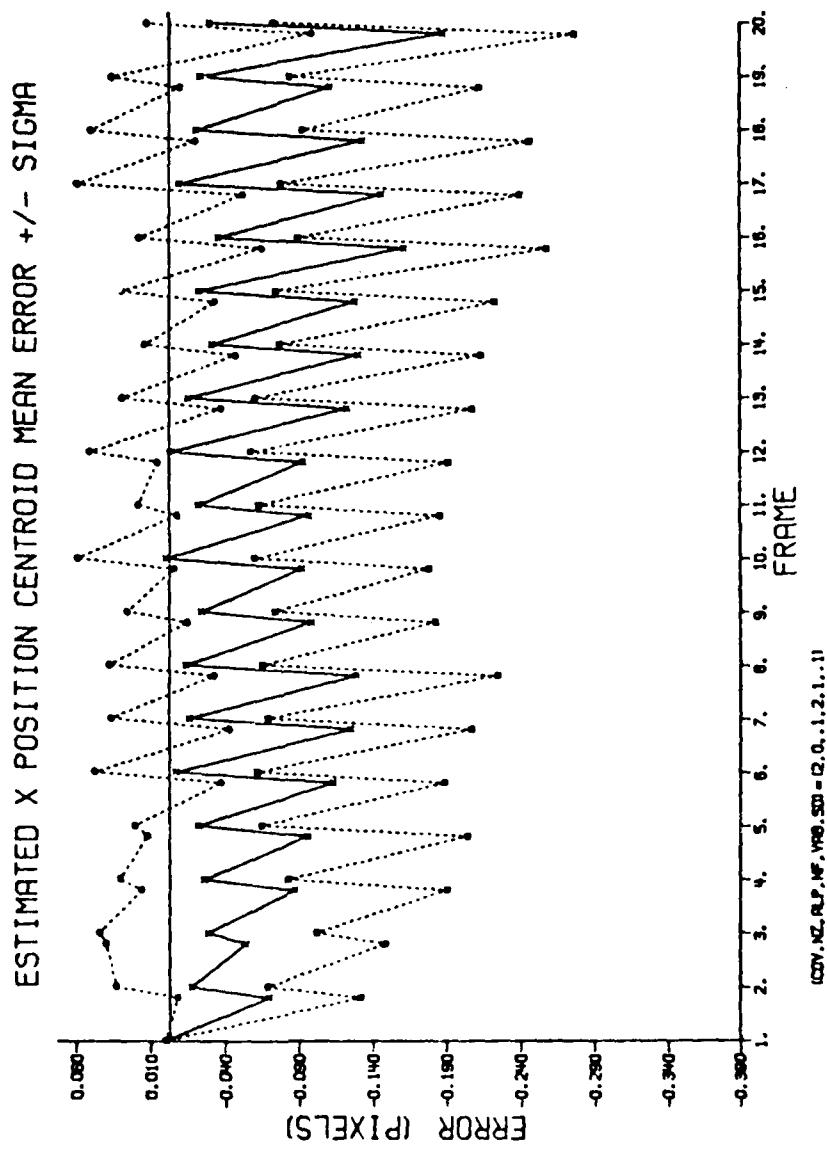


Figure 1-26. Removing Two Highest Spatial Frequencies
X Centroid Position Errors

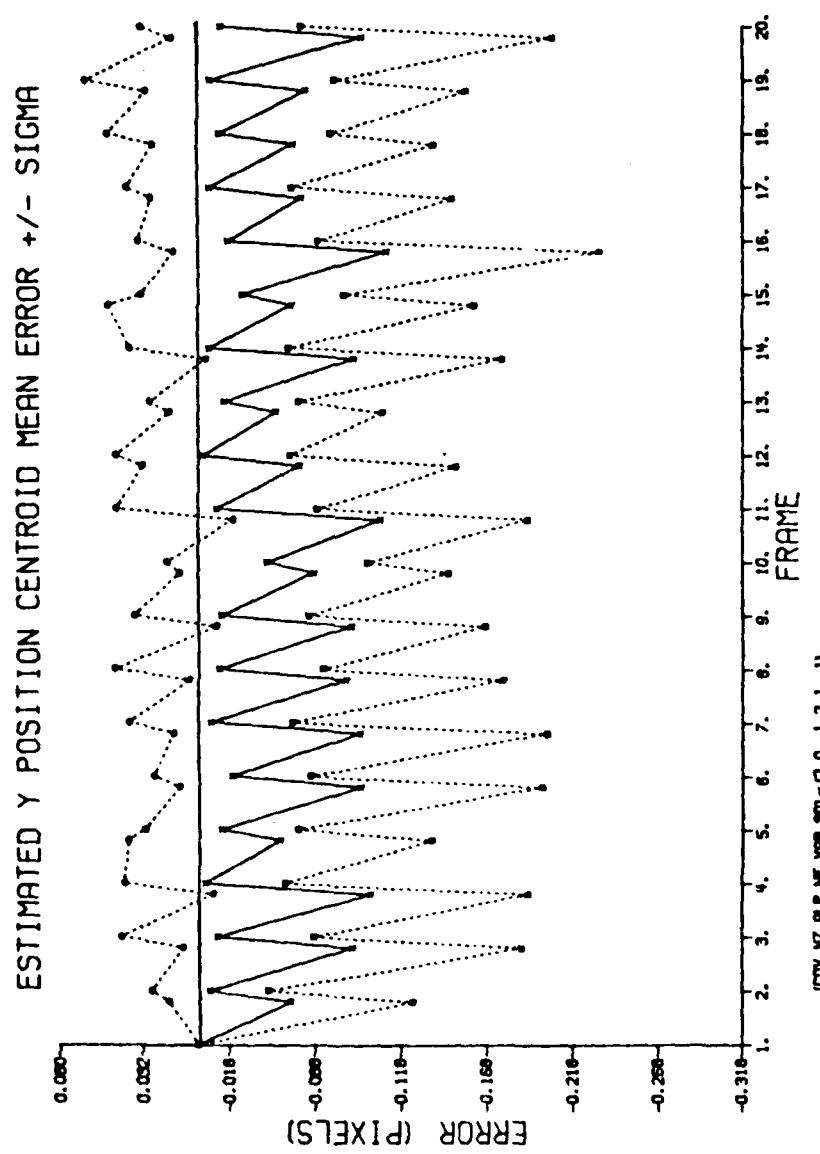


Figure 1-27. Removing Two Highest Spatial Frequencies
Y Centroid Position Errors

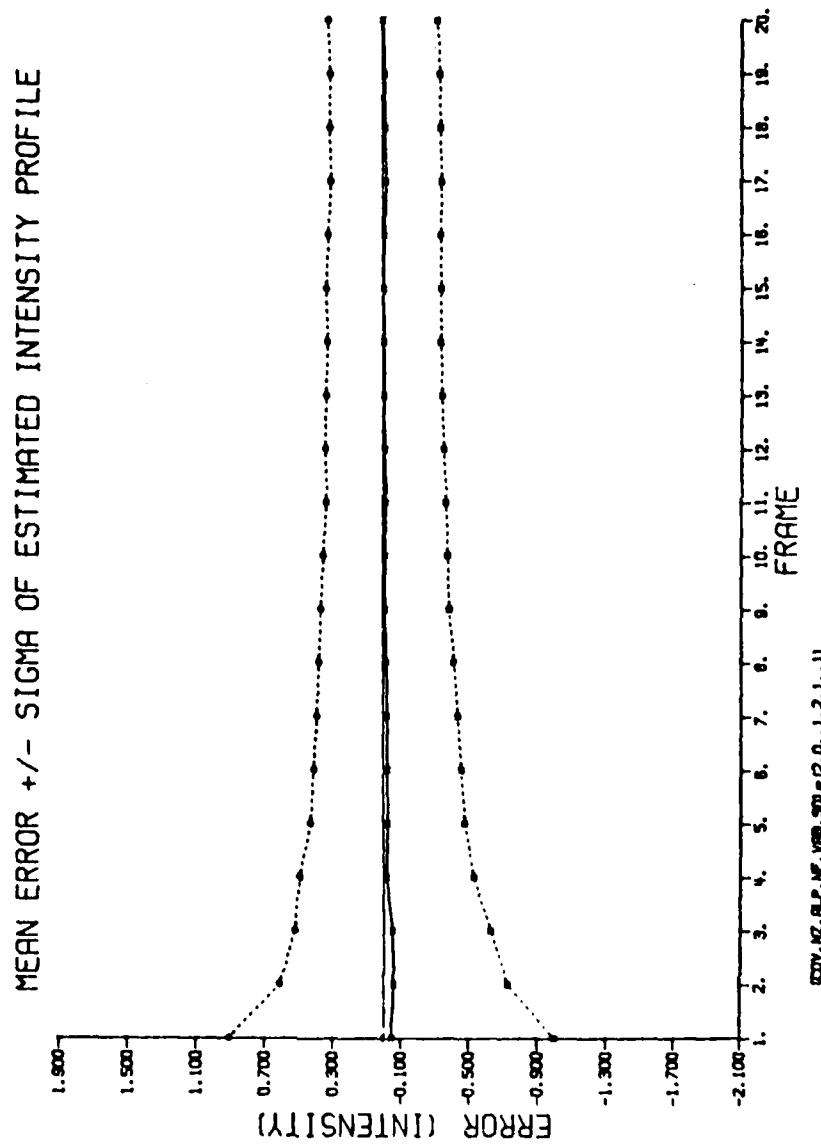


Figure 1-28. Removing Two Highest Spatial Frequencies
Error of Estimated \hat{h}

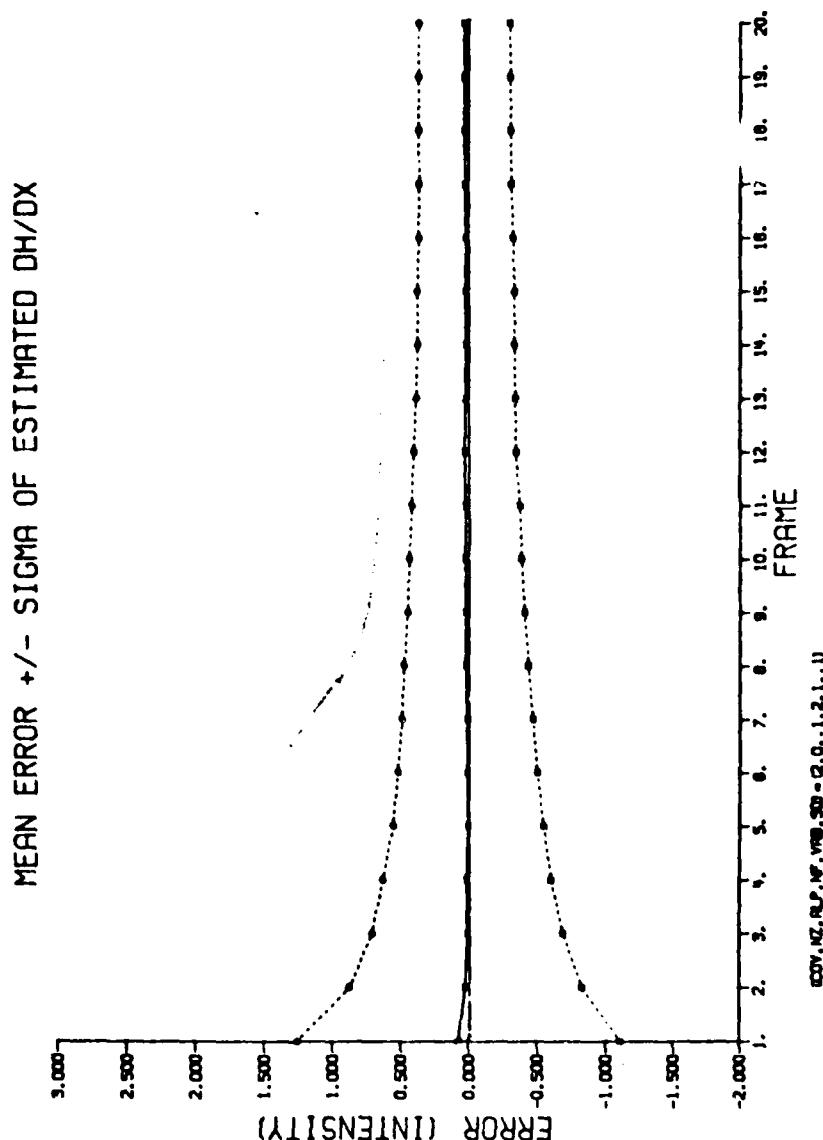


Figure 1-29. Removing Two Highest Spatial Frequencies
Error of Estimated Dh/Dx

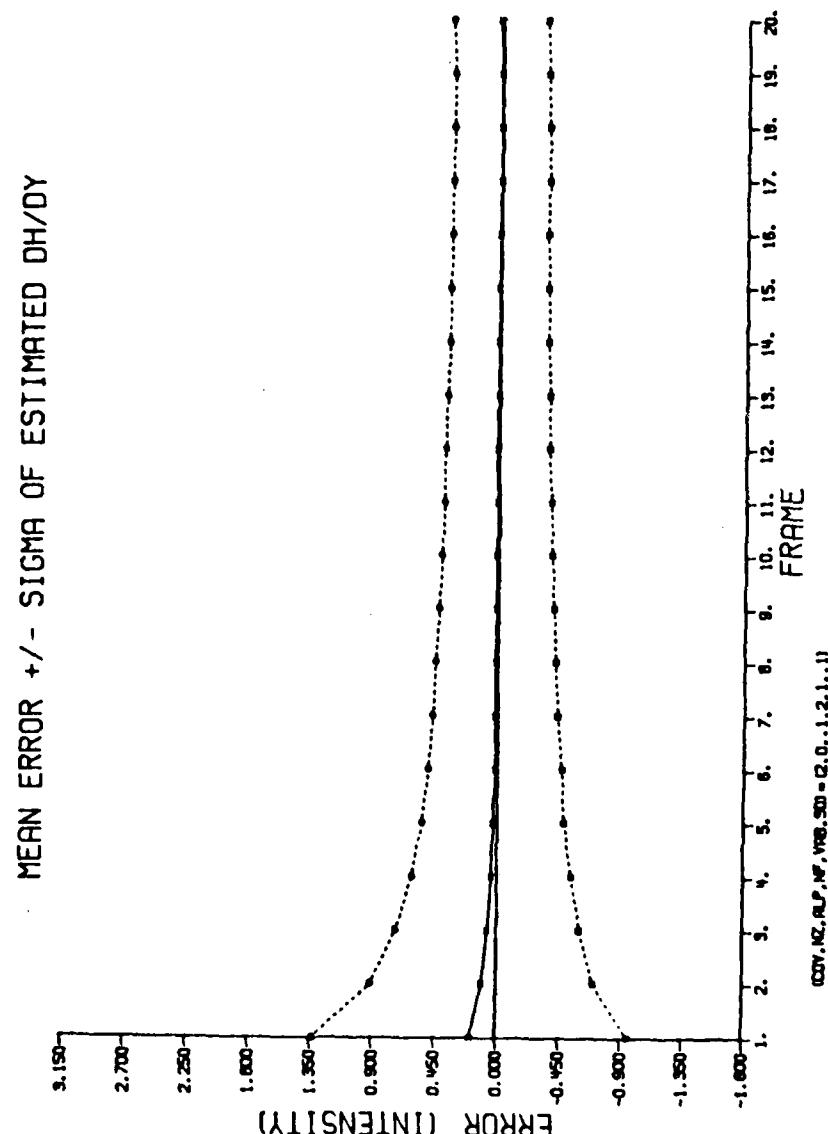


Figure 1-30. Removing Two Highest Spatial Frequencies
Error of Estimated Dh/Dy

Removing Four Highest
Spatial Frequencies

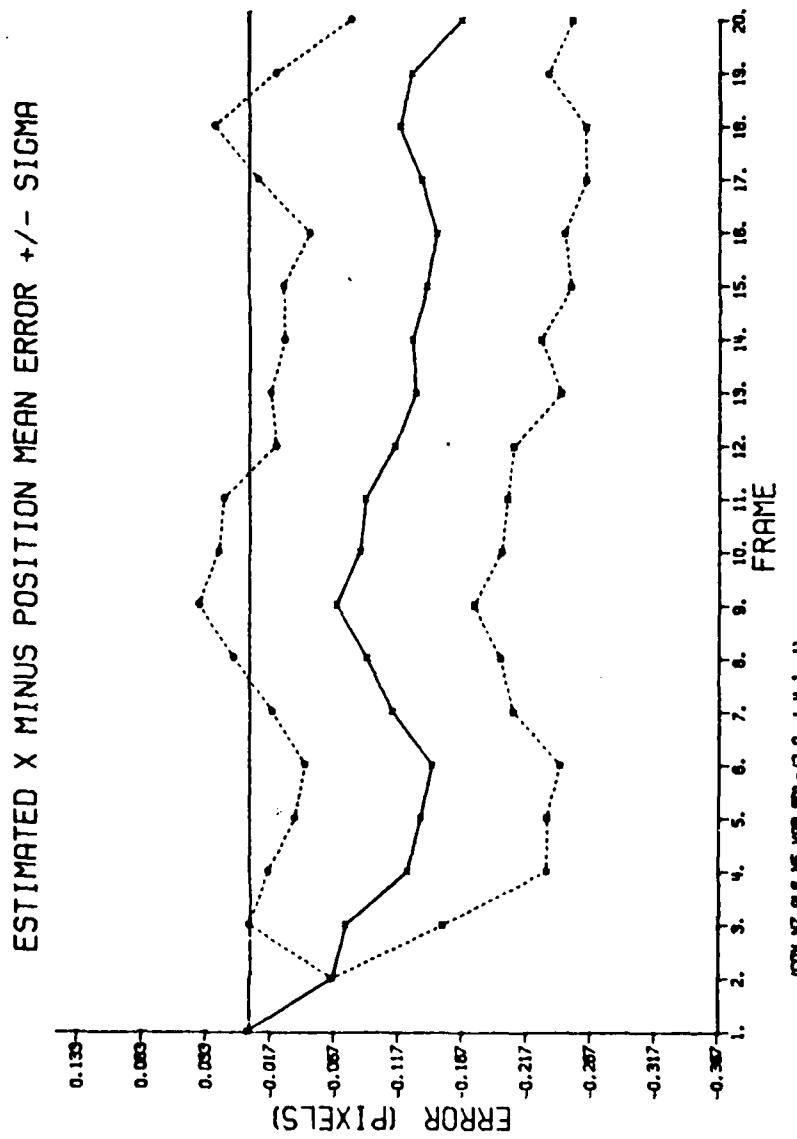


Figure 1-31. Removing Four Highest Spatial Frequencies
X Minus Errors

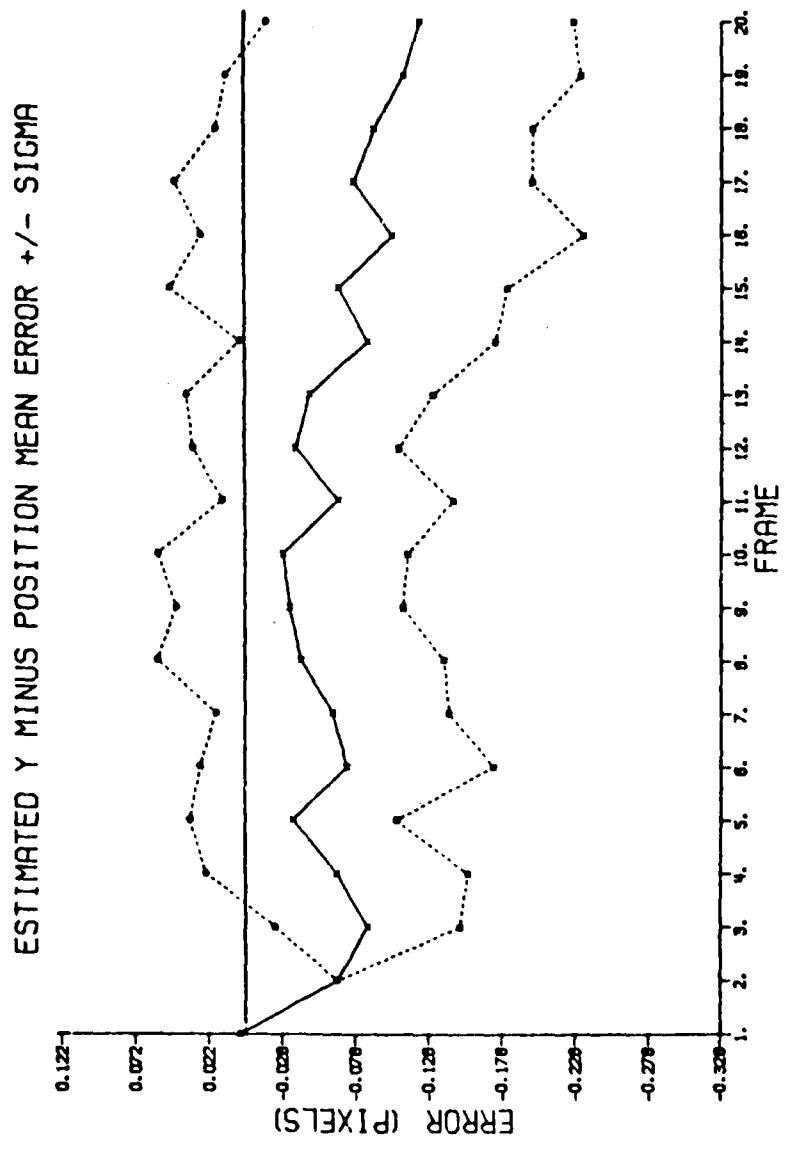


Figure 1-32. Removing Four Highest Spatial Frequencies
Y Minus Errors

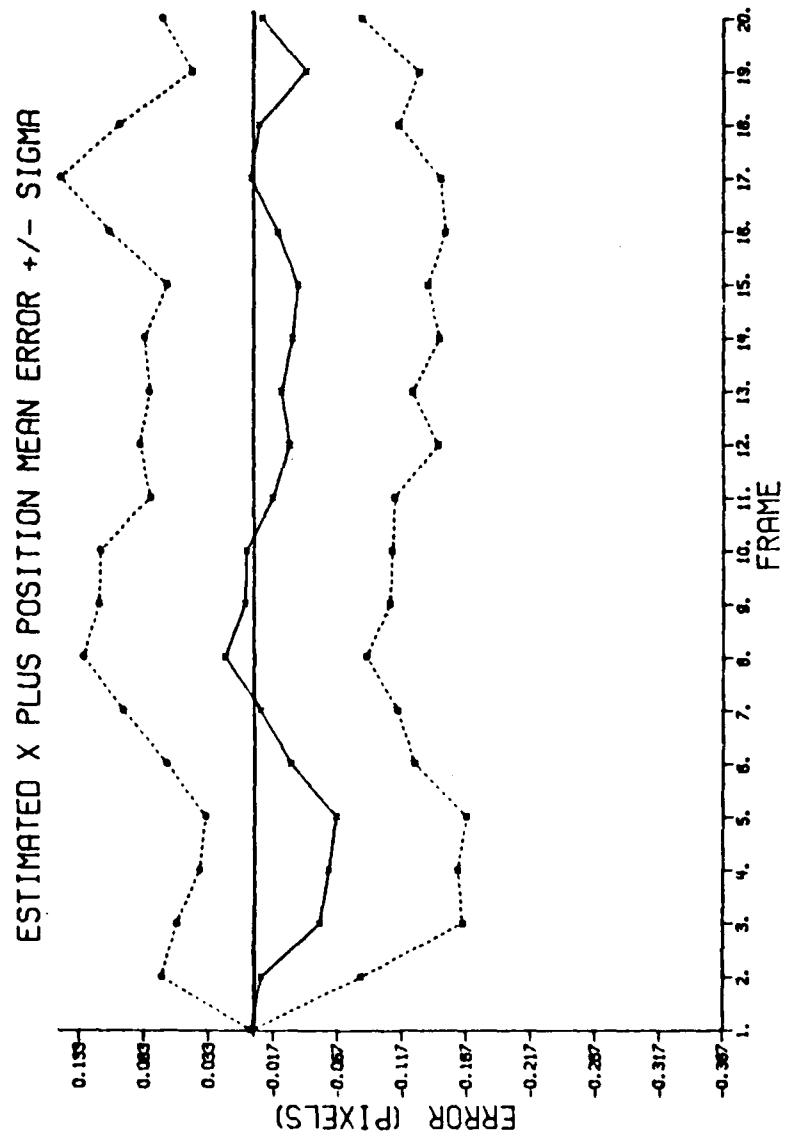


Figure 1-33. Removing Four Highest Spatial Frequencies
X Plus Errors

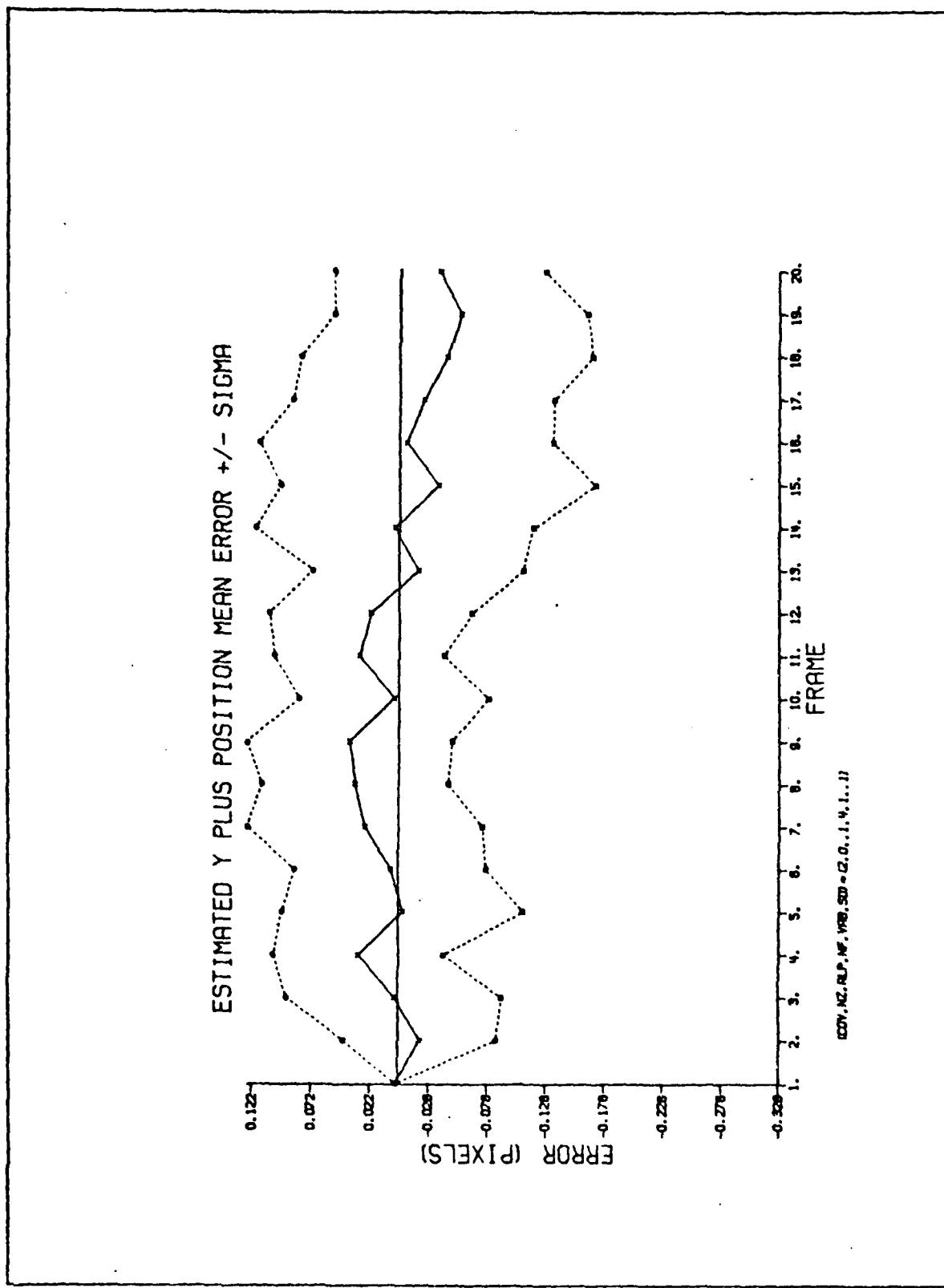


Figure 1-34. Removing Four Highest Spatial Frequencies
Y Plus Errors

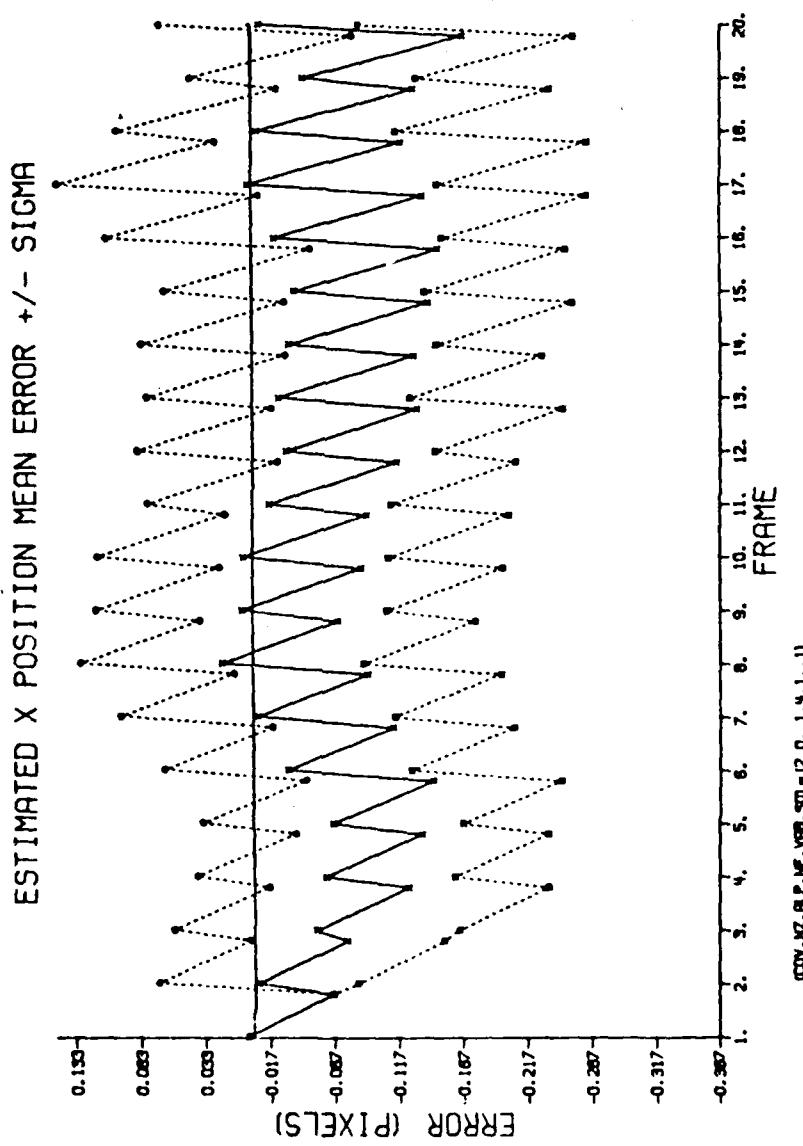


Figure 1-35. Removing Four Highest Spatial Frequencies
X Position Errors

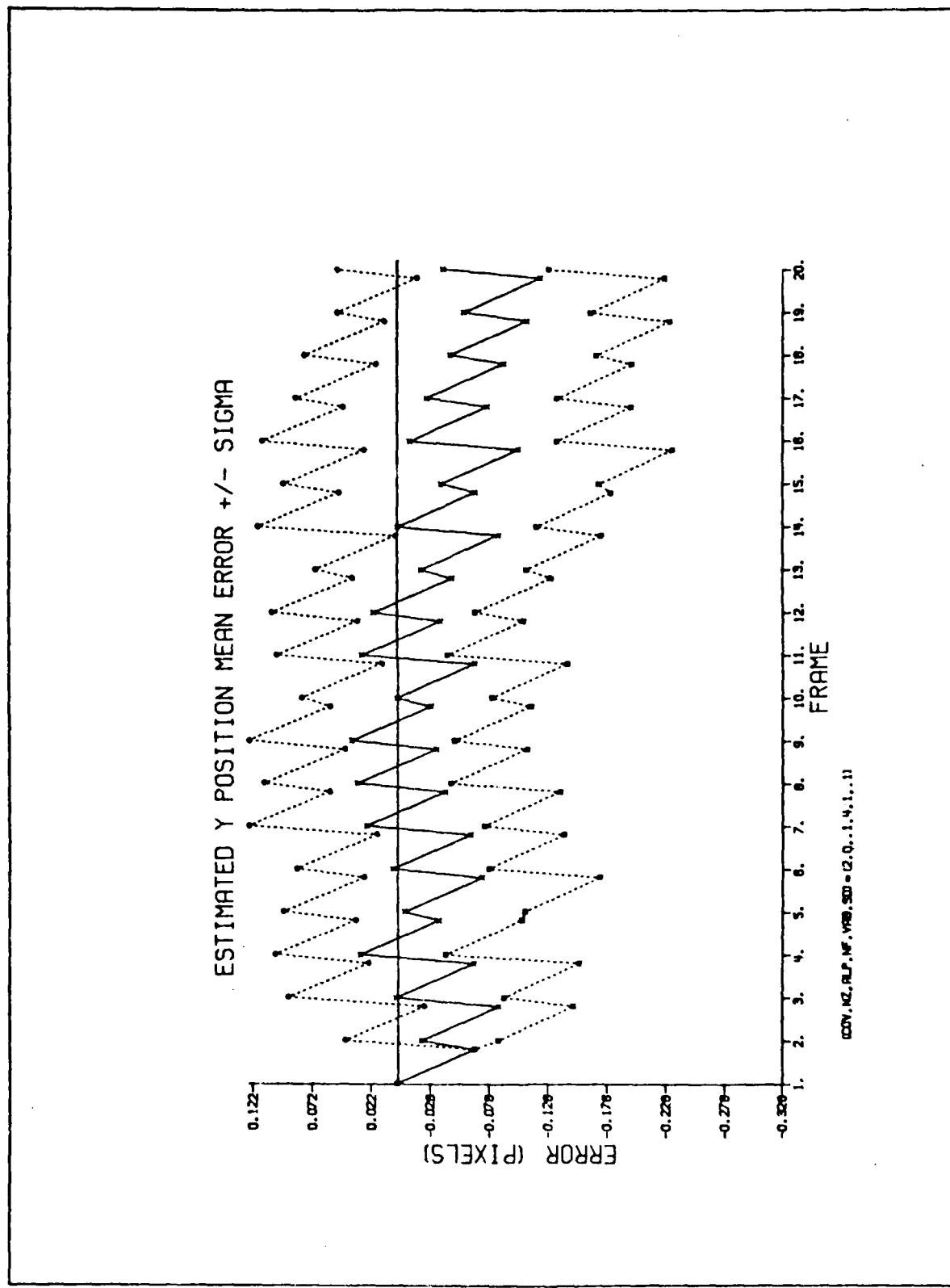


Figure 1-36. Removing Four Highest Spatial Frequencies
Y Position Errors

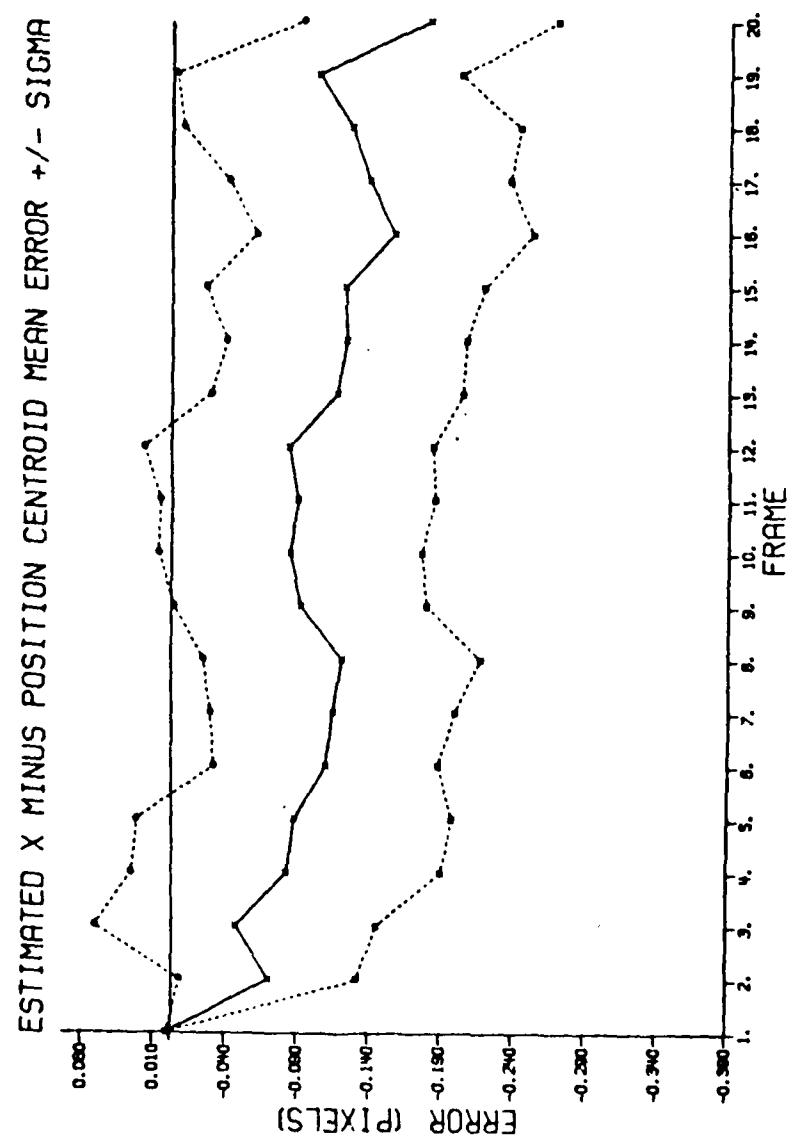


Figure 1-37. Removing Four Highest Spatial Frequencies
X Centroid Minus Errors

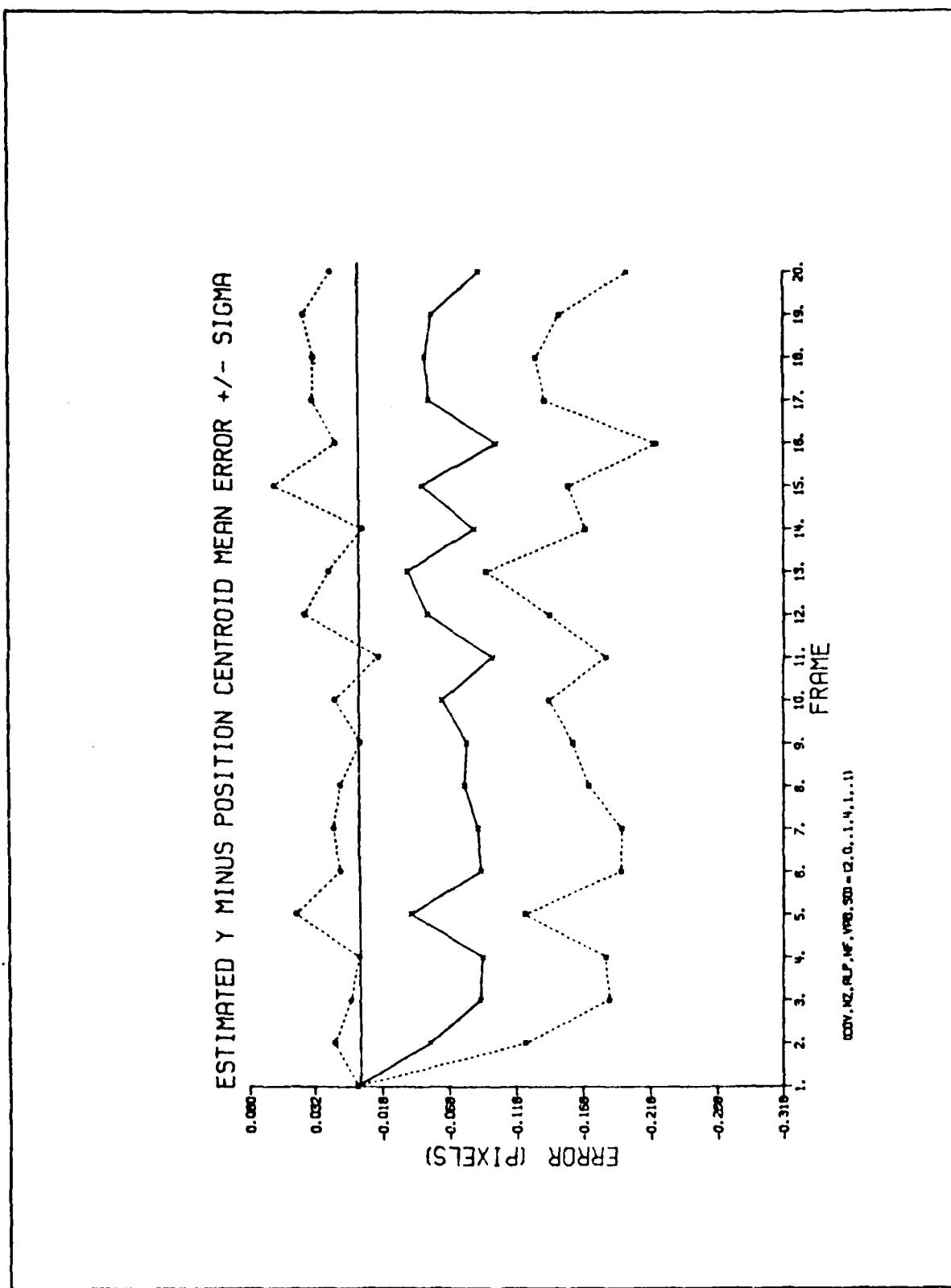


Figure 1-38. Removing Four Highest Spatial Frequencies
 γ Centroid Minus Errors

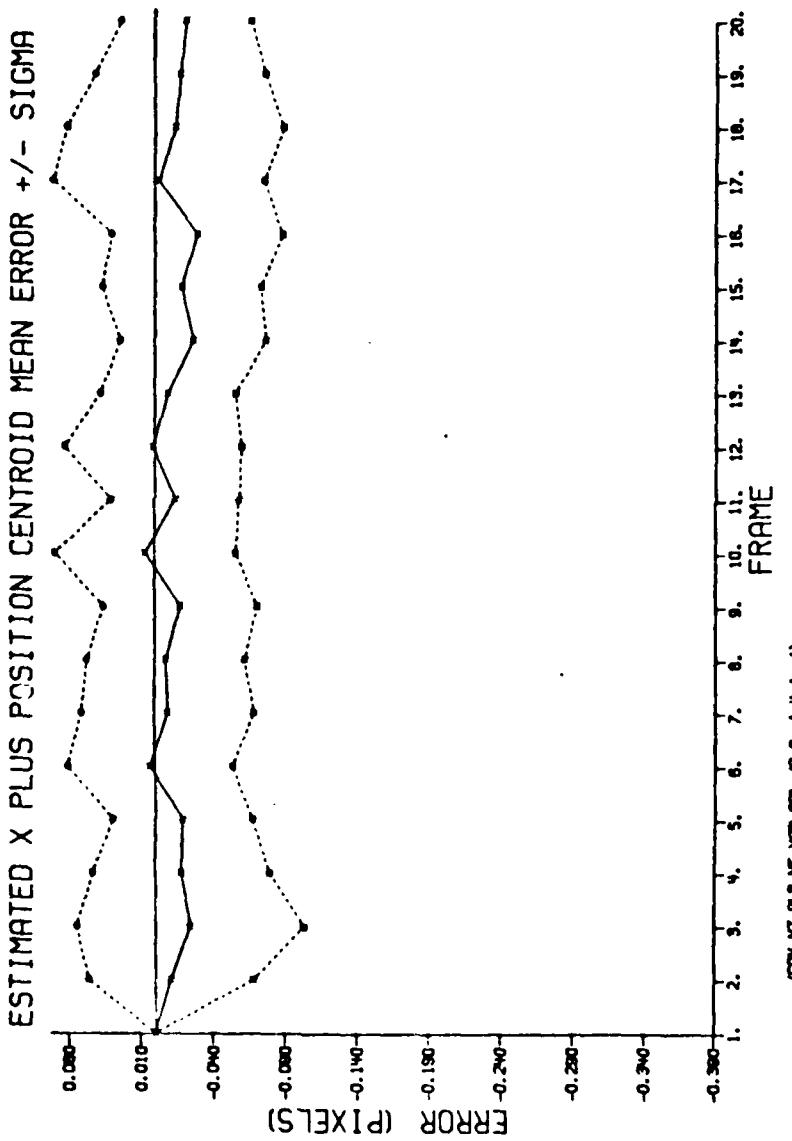


Figure 1-39. Removing Four Highest Spatial Frequencies
 X Centroid Plus Errors

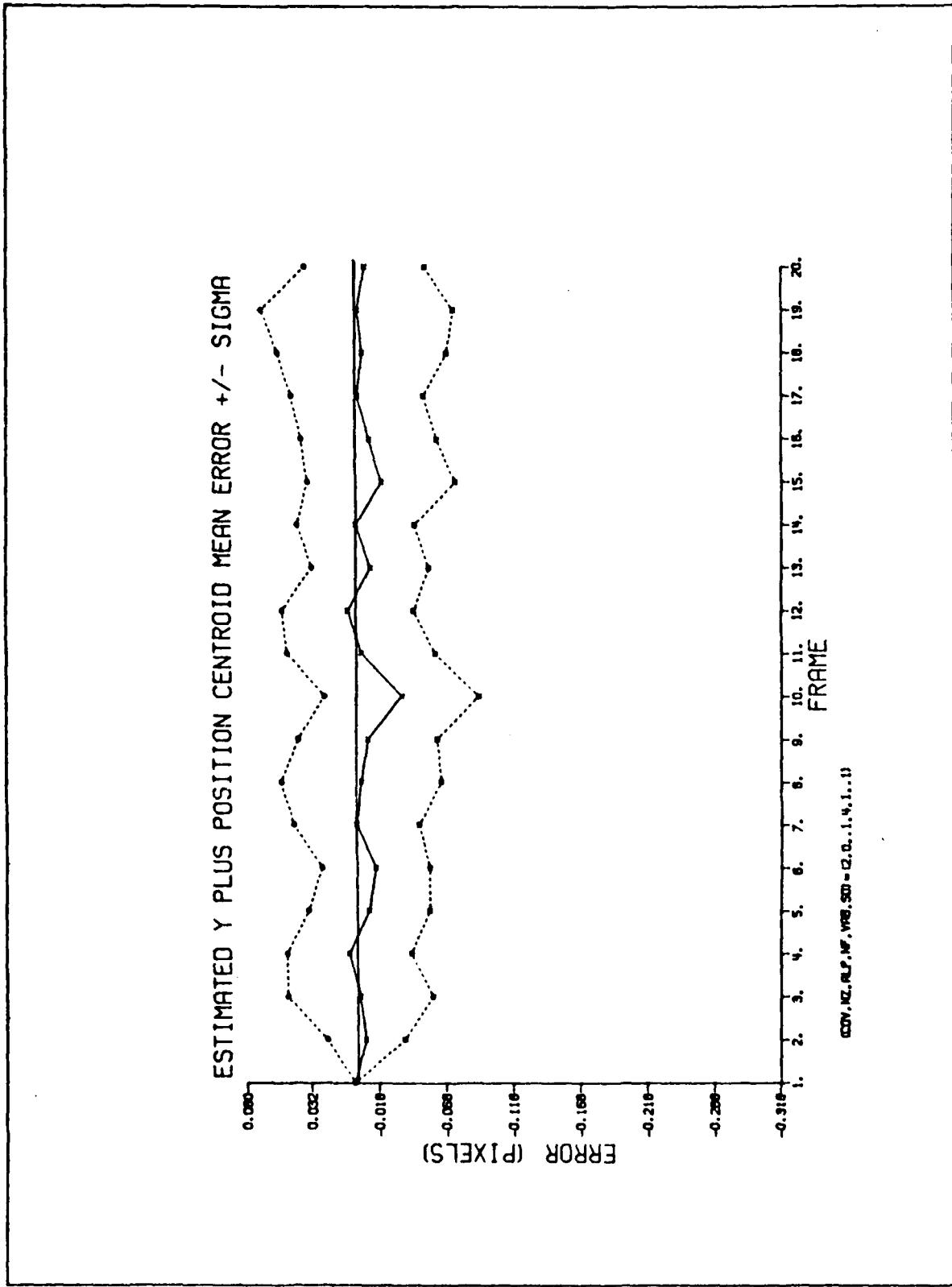


Figure 1-40. Removing Four Highest Spatial Frequencies
Y Centroid Plus Errors

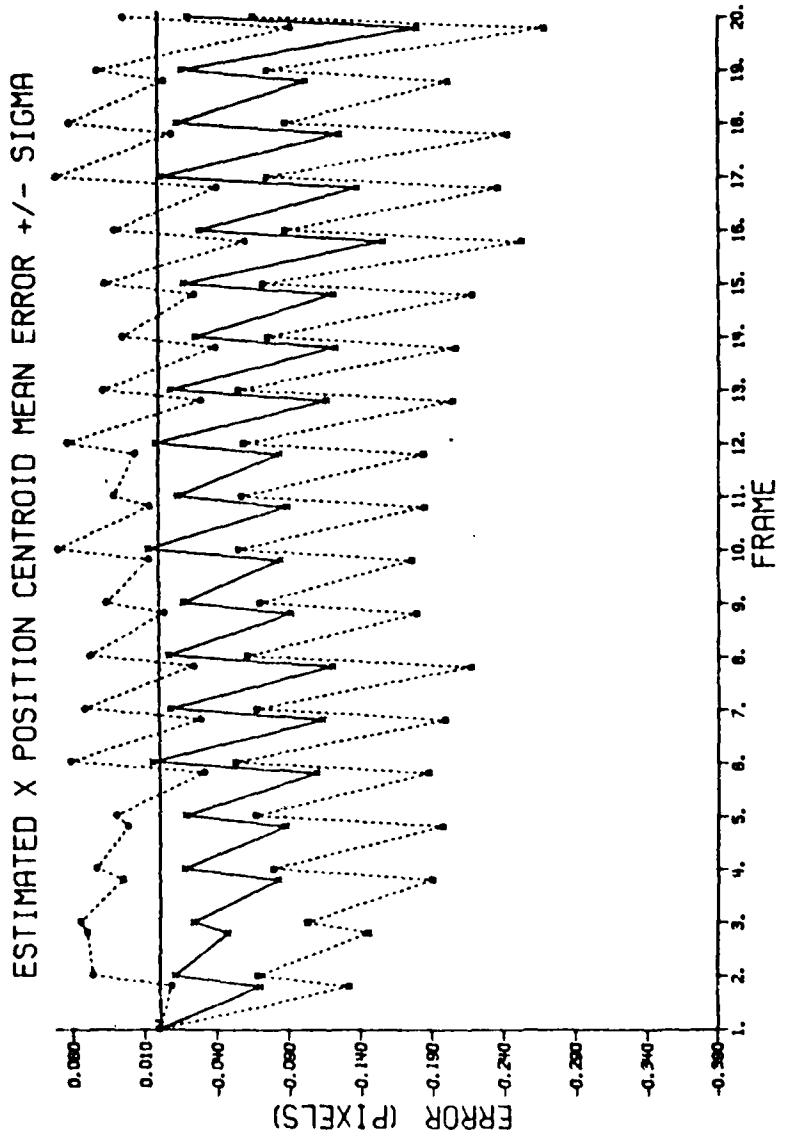


Figure 1-41. Removing Four Highest Spatial Frequencies
X Centroid Position Errors

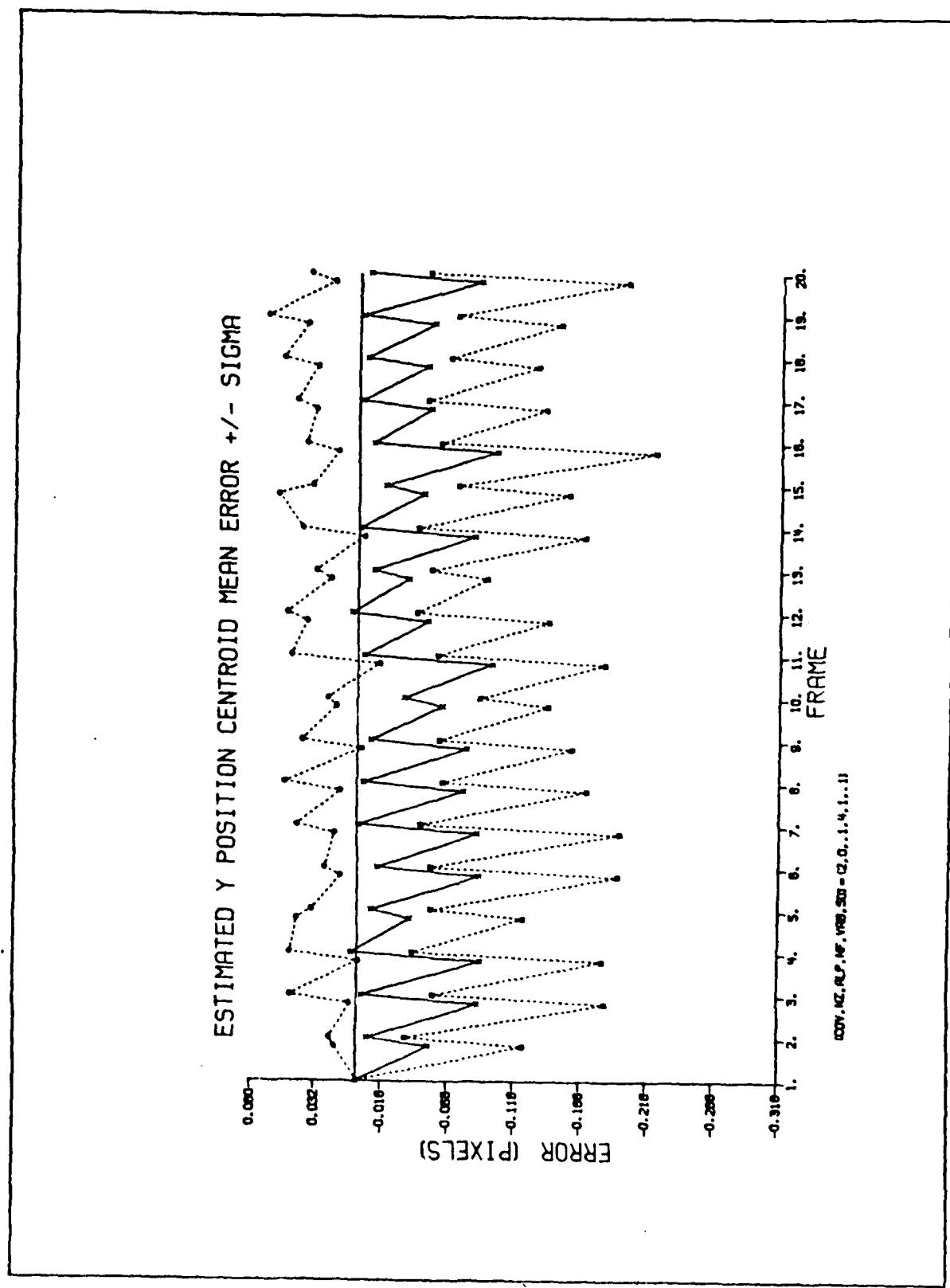


Figure 1-42. Removing Four Highest Spatial Frequencies
Y Centroid Position Errors

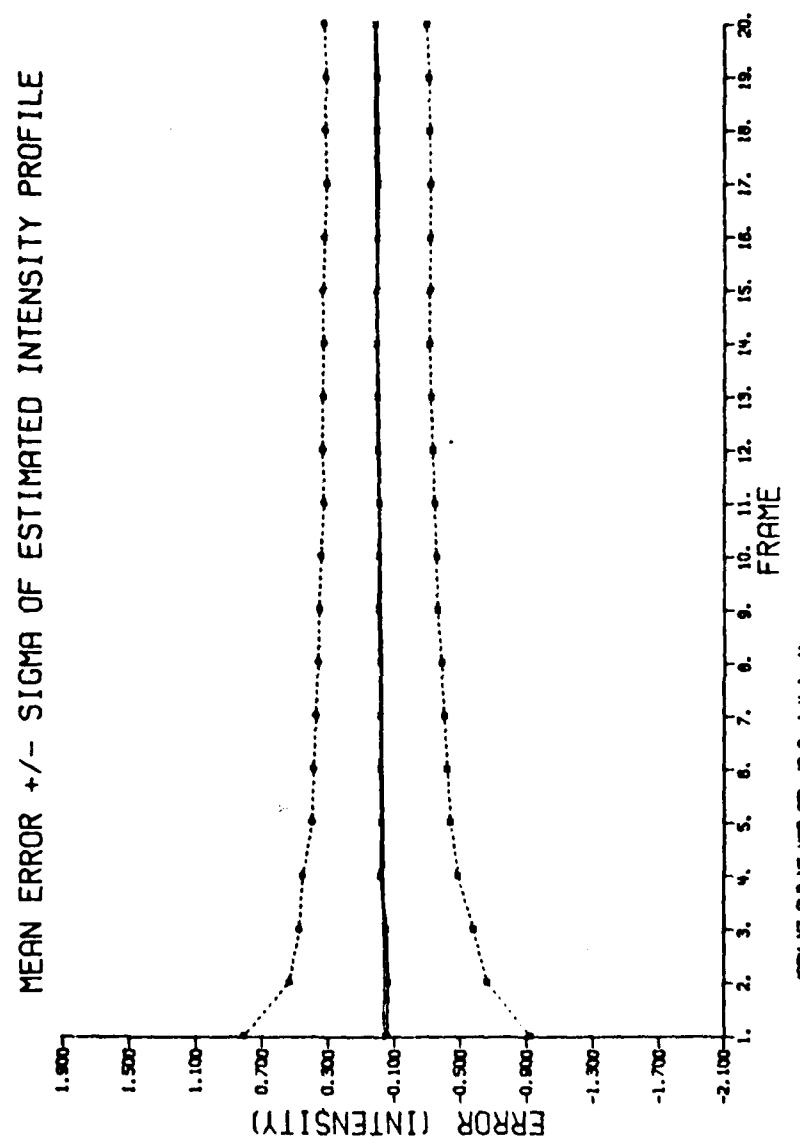


Figure 1-43. Removing Four Highest Spatial Frequencies
Error of Estimated \hat{h}

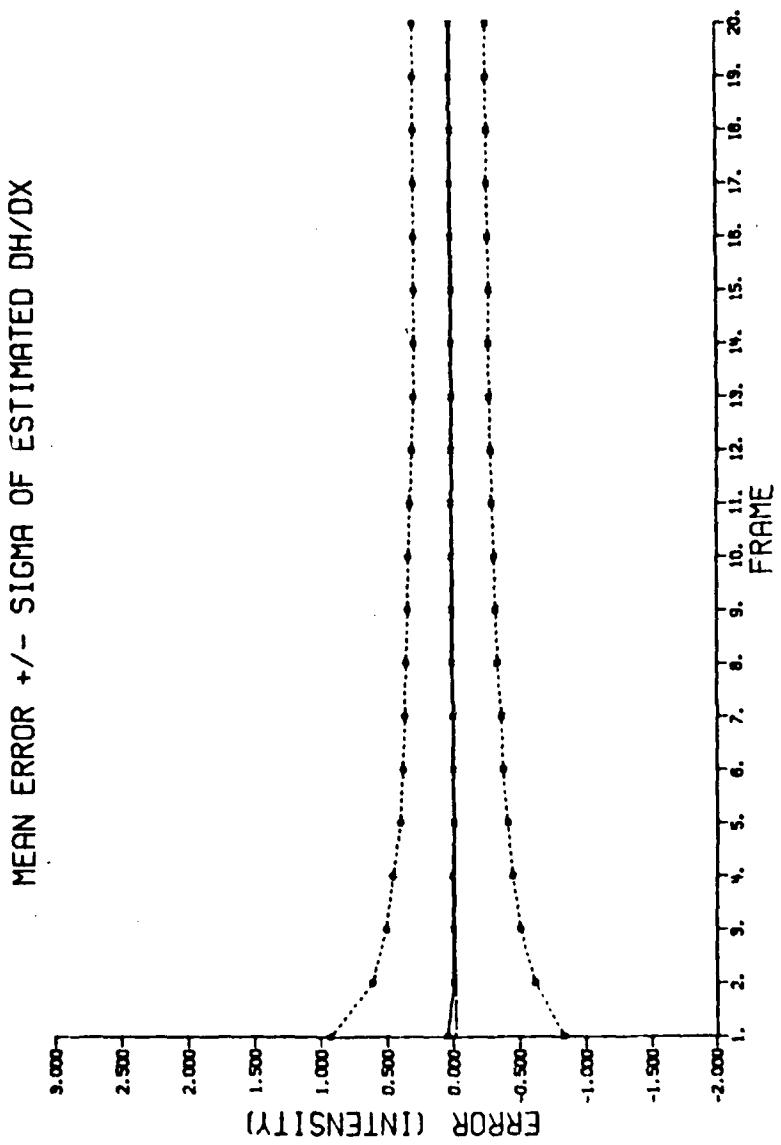


Figure 1-44. Removing Four Highest Spatial Frequencies
Error of Estimated Dh/Dx

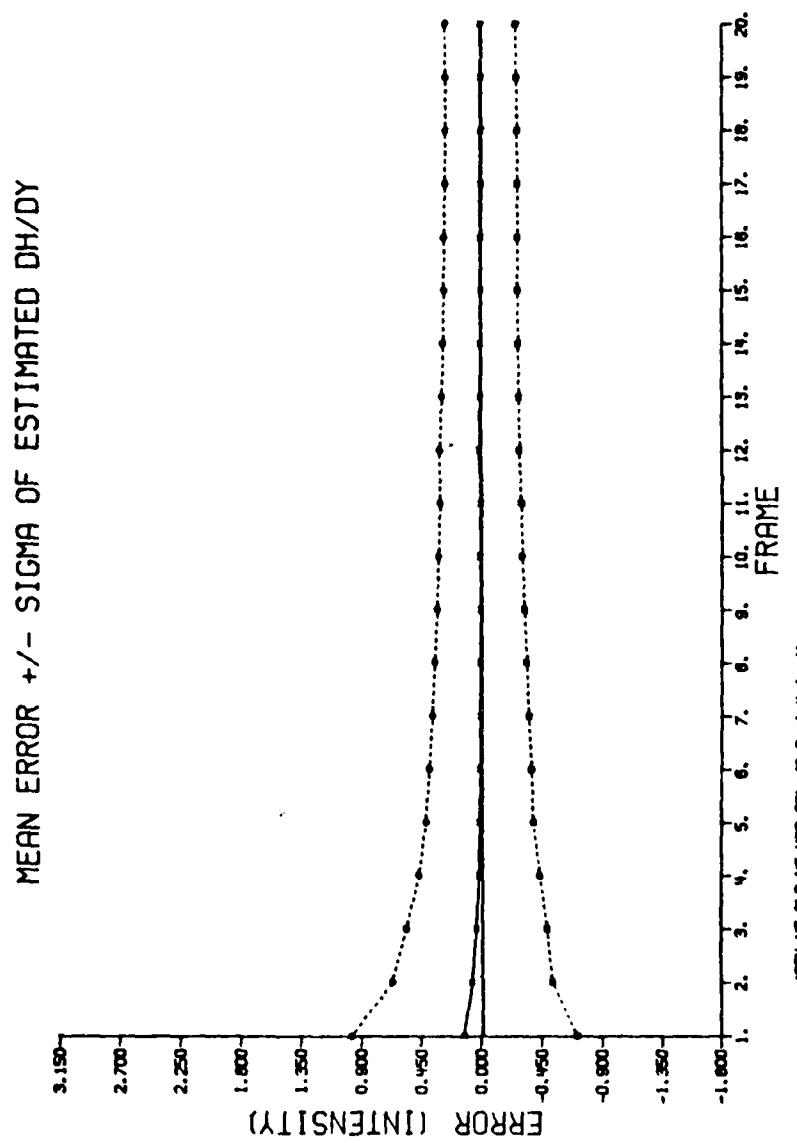


Figure 1-45. Removing Four Highest Spatial Frequencies
Error of Estimated Dh/Dy

Maximum Noise

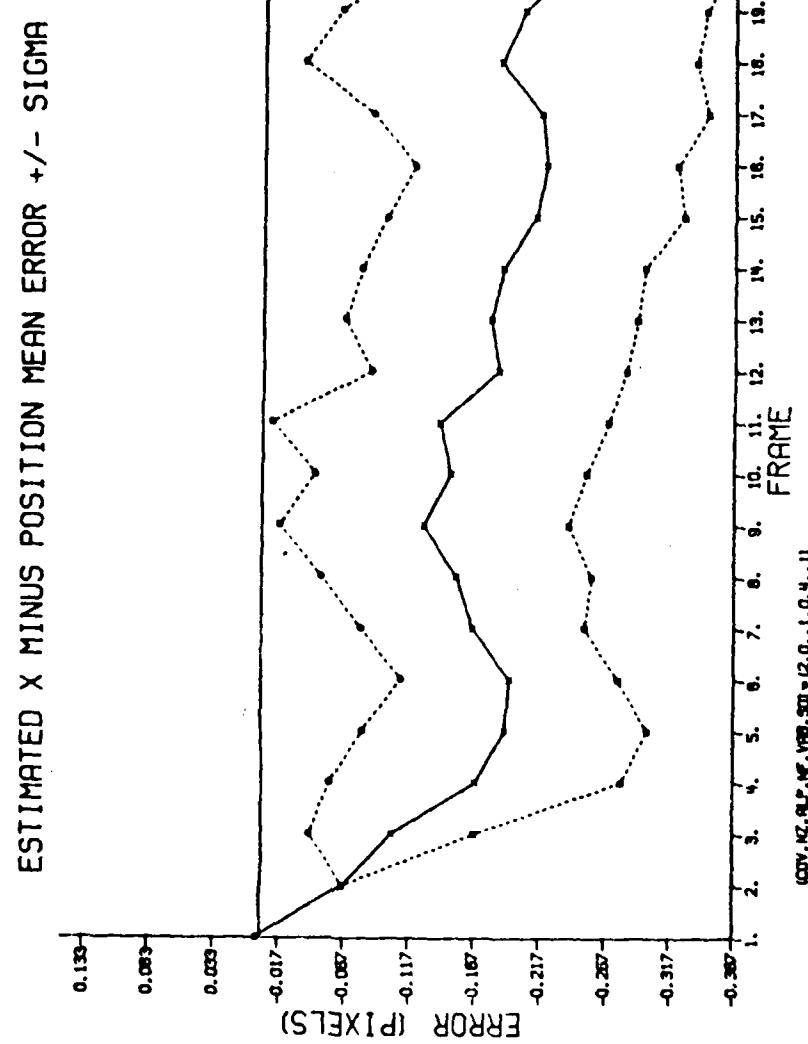


Figure 1-46. Maximum Noise X Minus Errors

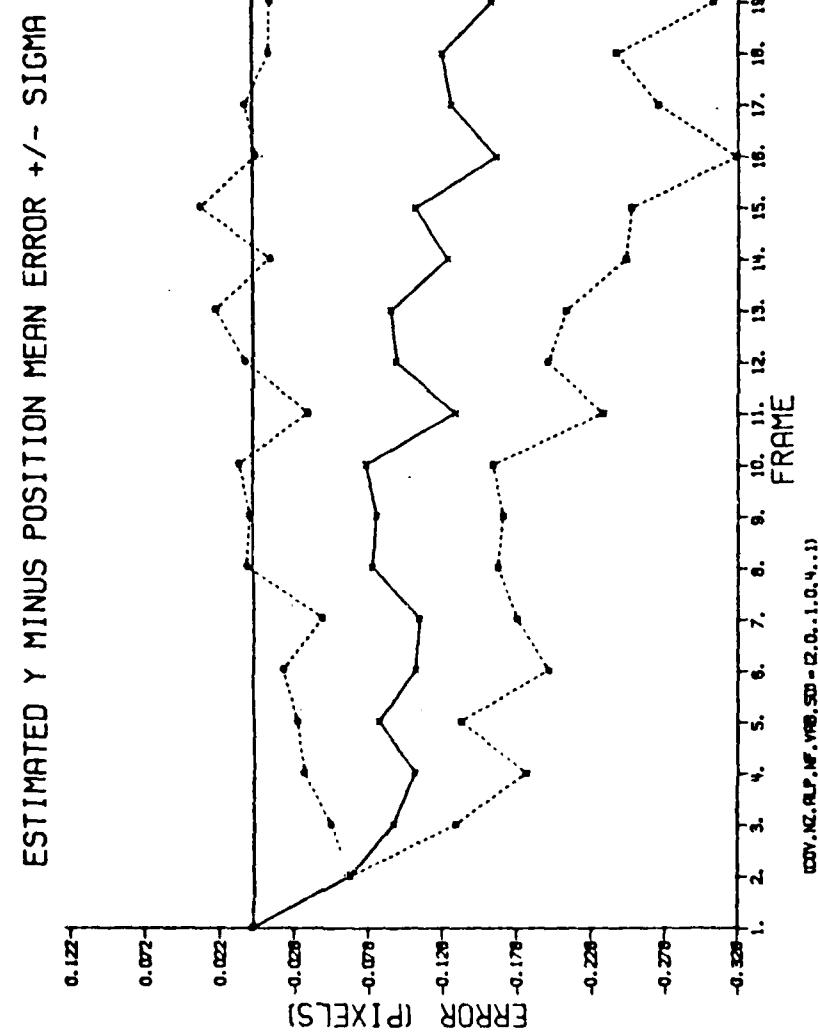


Figure 1-47. Maximum Noise Y Minus Errors

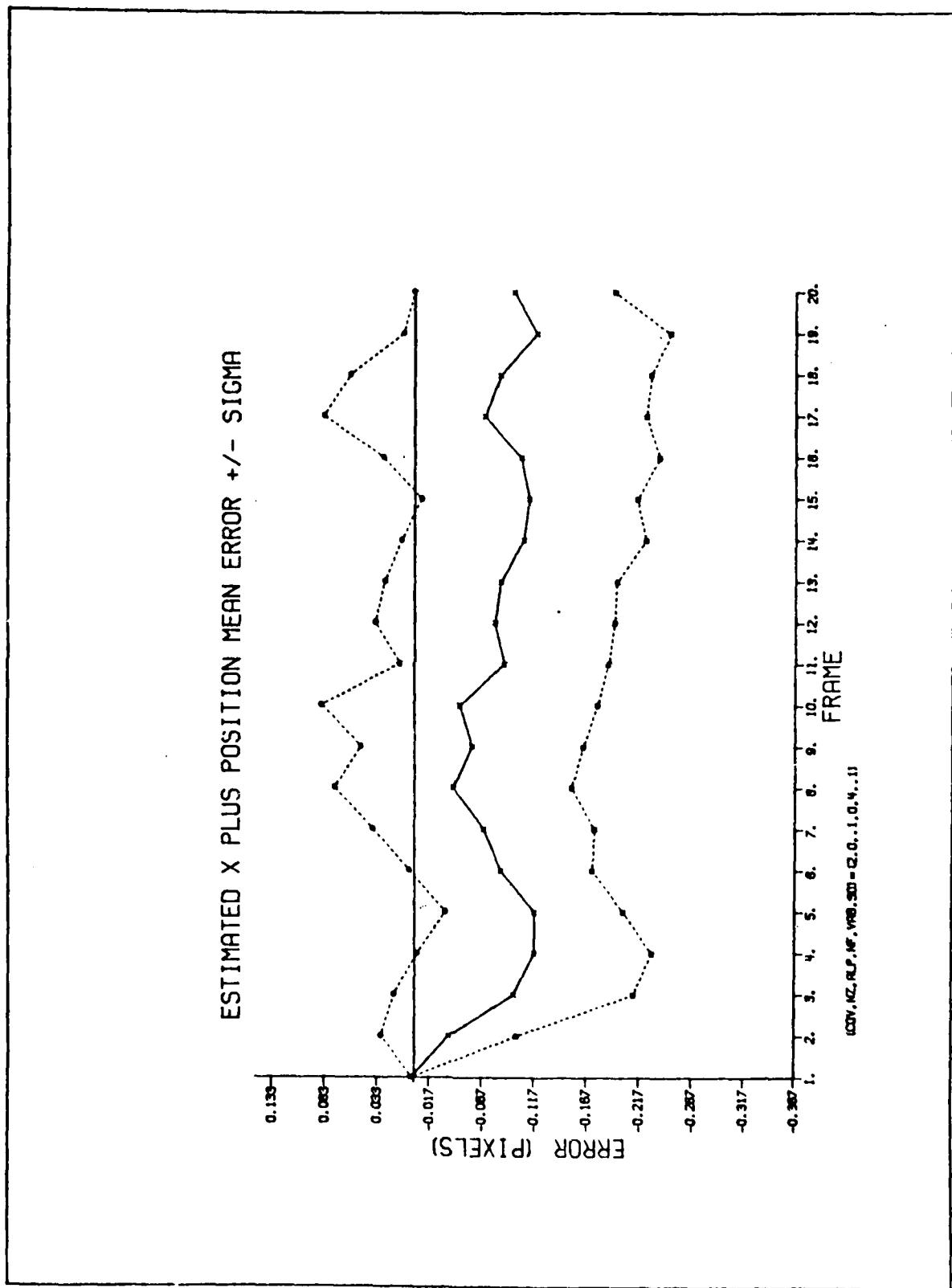


Figure 1-48. Maximum Noise X Plus Errors

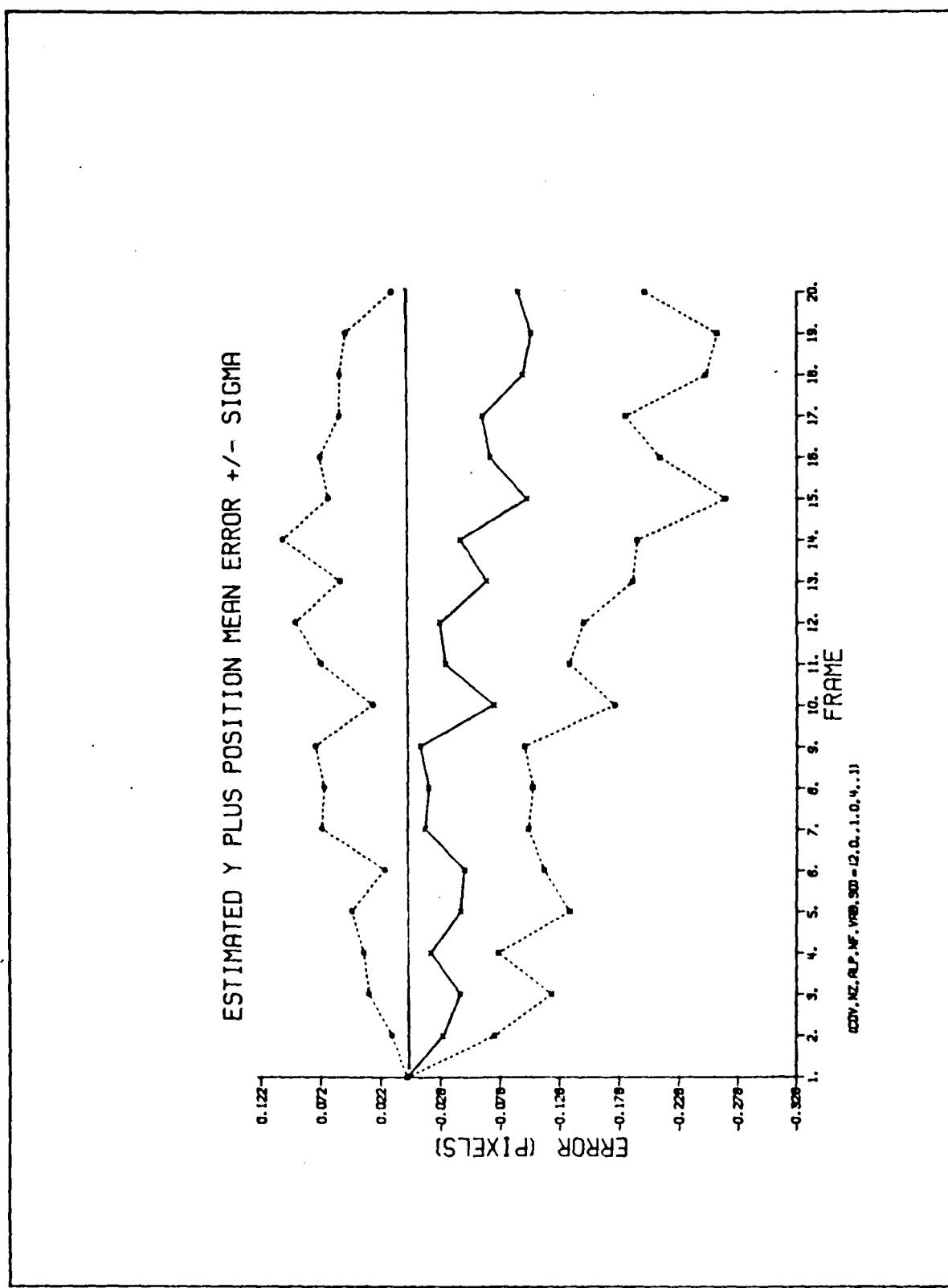


Figure 1-49. Maximum Noise Y Plus Errors

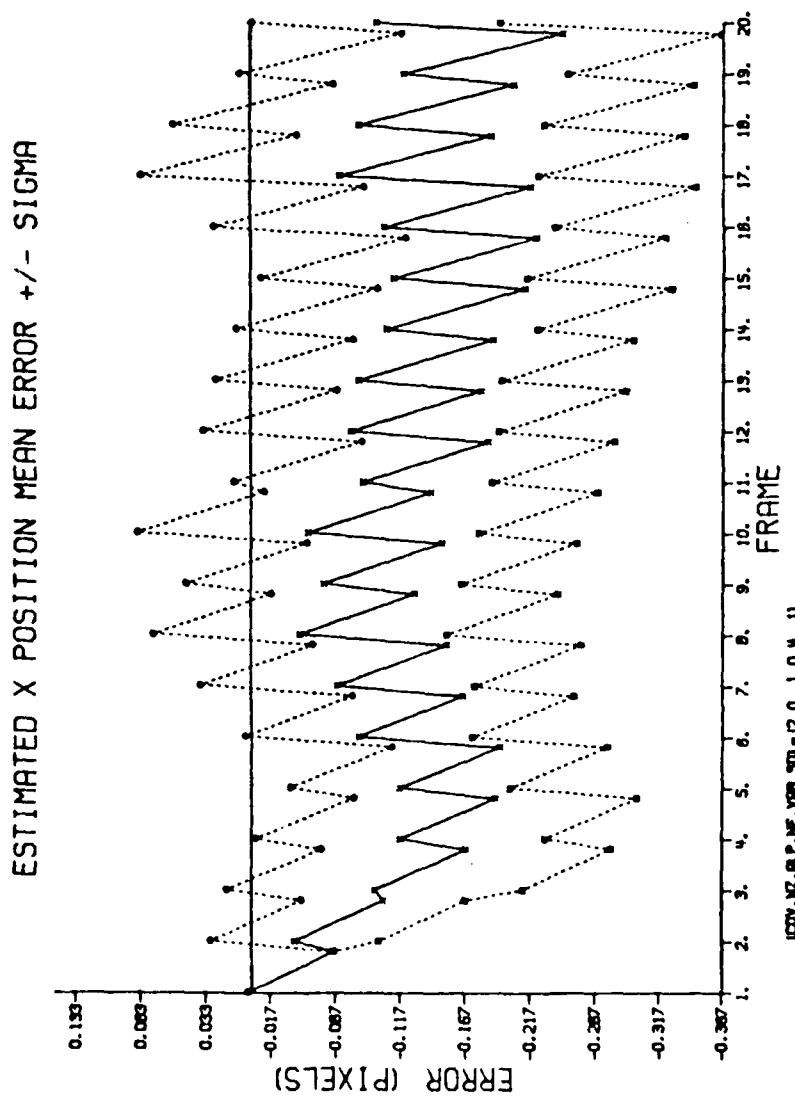


Figure 1-50. Maximum Noise X Position Errors

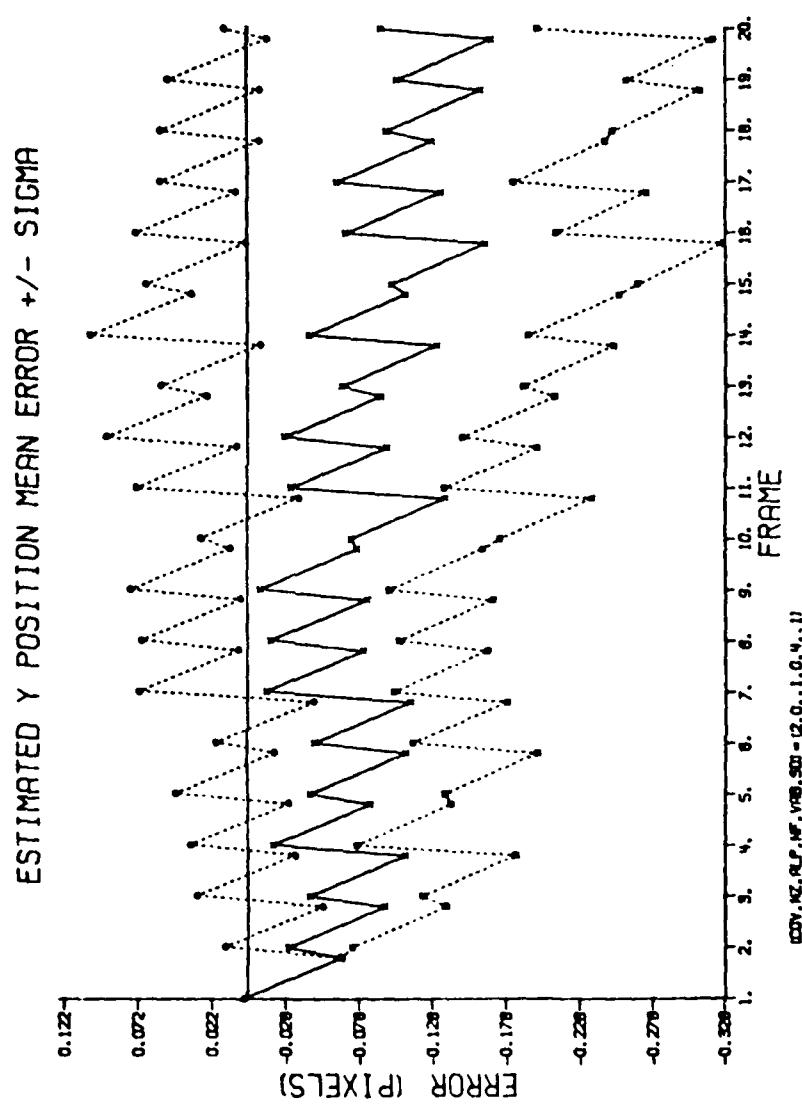


Figure 1-51. Maximum Noise Y Position Errors

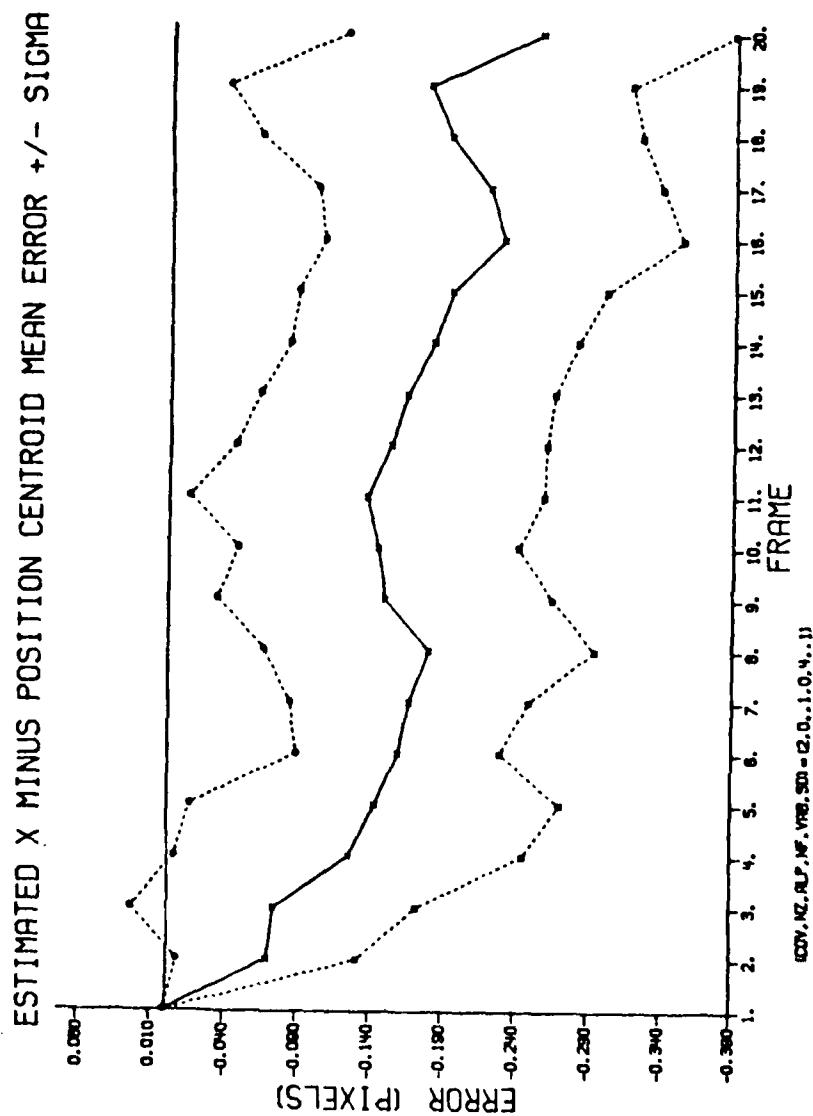


Figure 1-52. Maximum Noise X Centroid Minus Errors

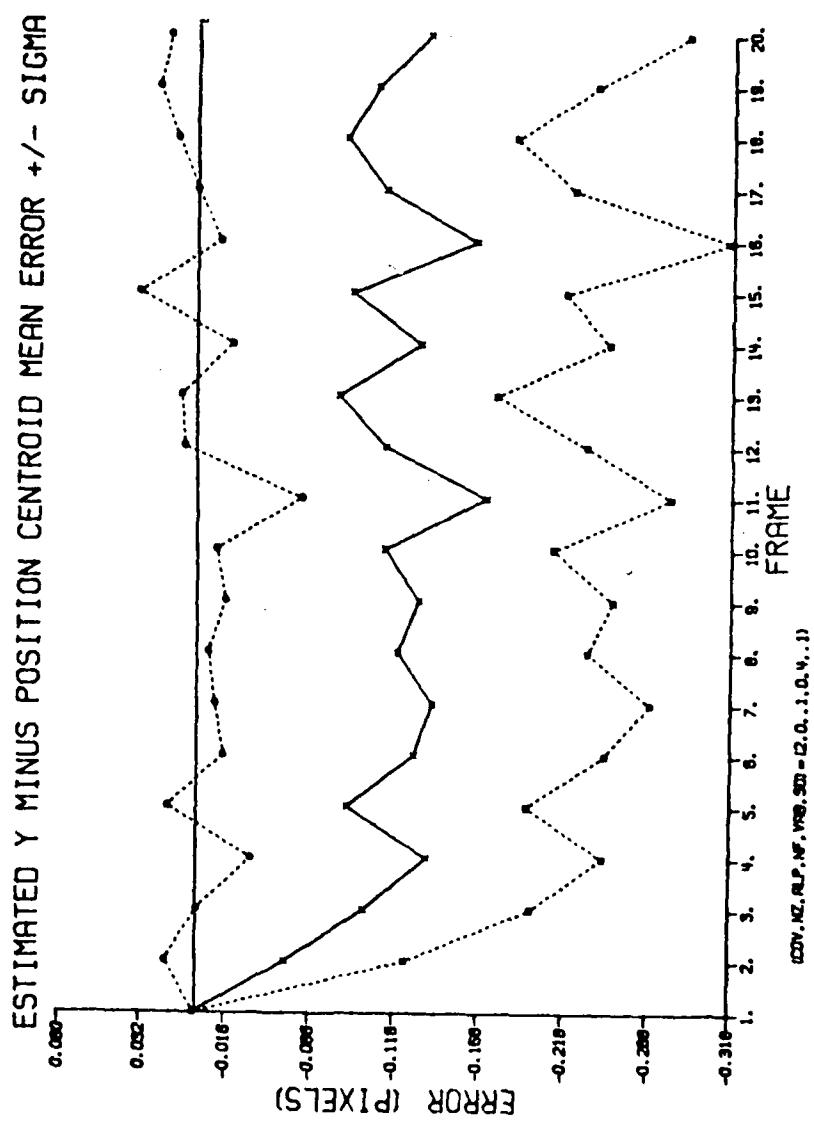


Figure 1-53. Maximum Noise Y Centroid Minus Errors

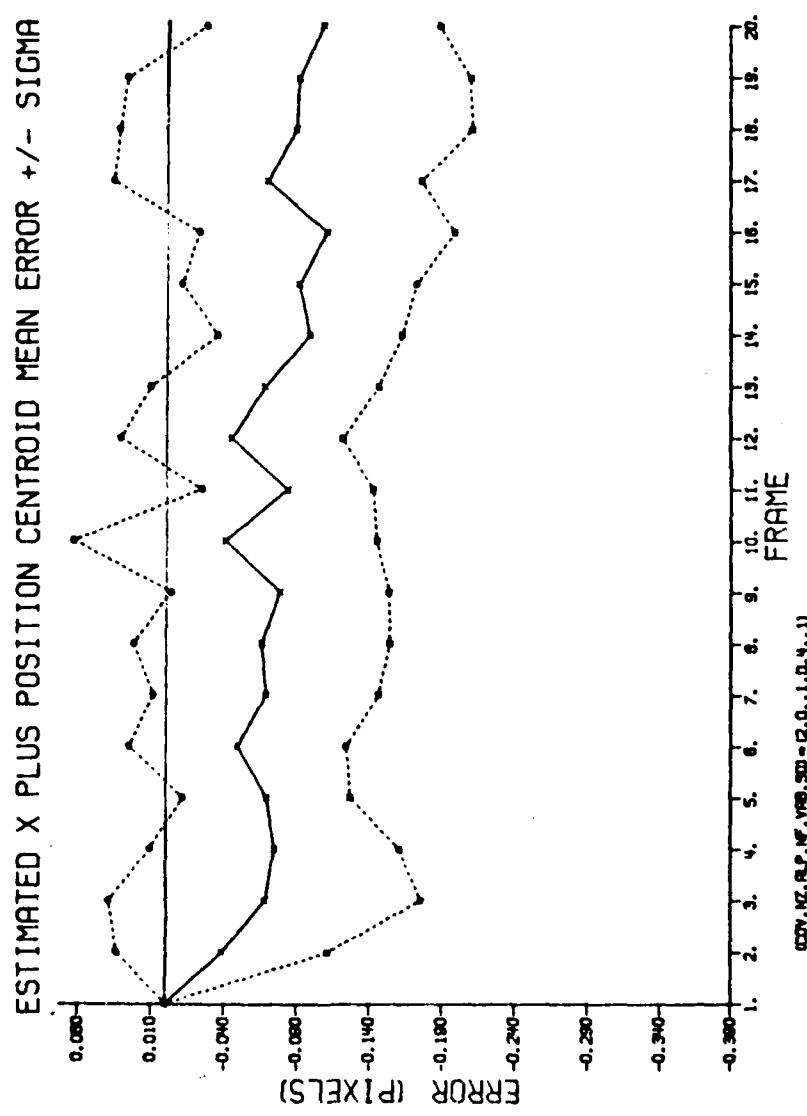


Figure 1-54. Maximum Noise X Centroid Plus Errors

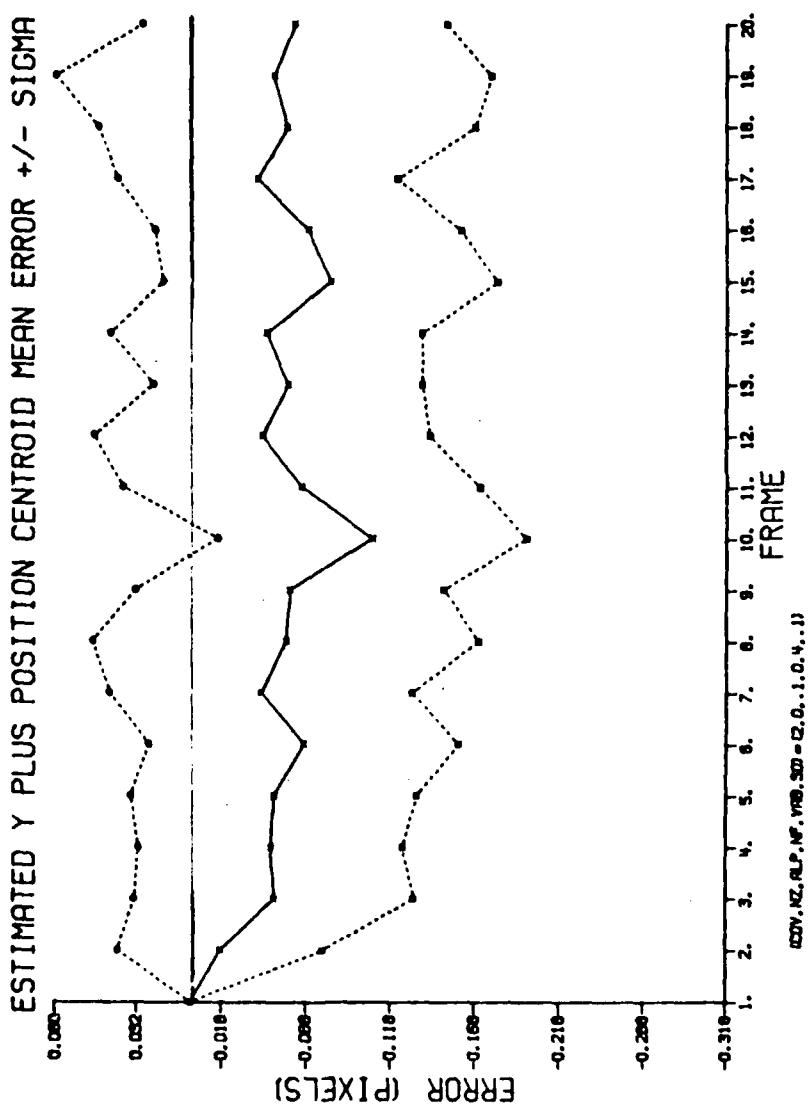


Figure 1-55. Maximum Noise Y Centroid Plus Errors

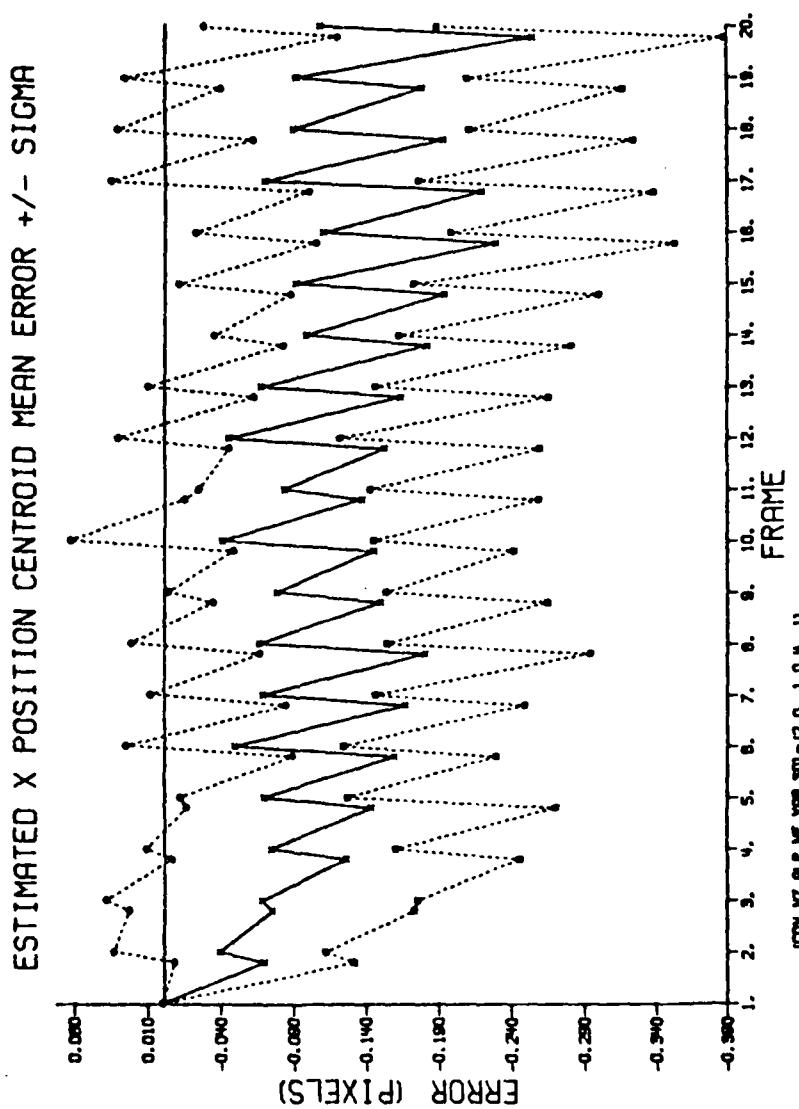


Figure 1-56. Maximum Noise X Centroid Position Errors

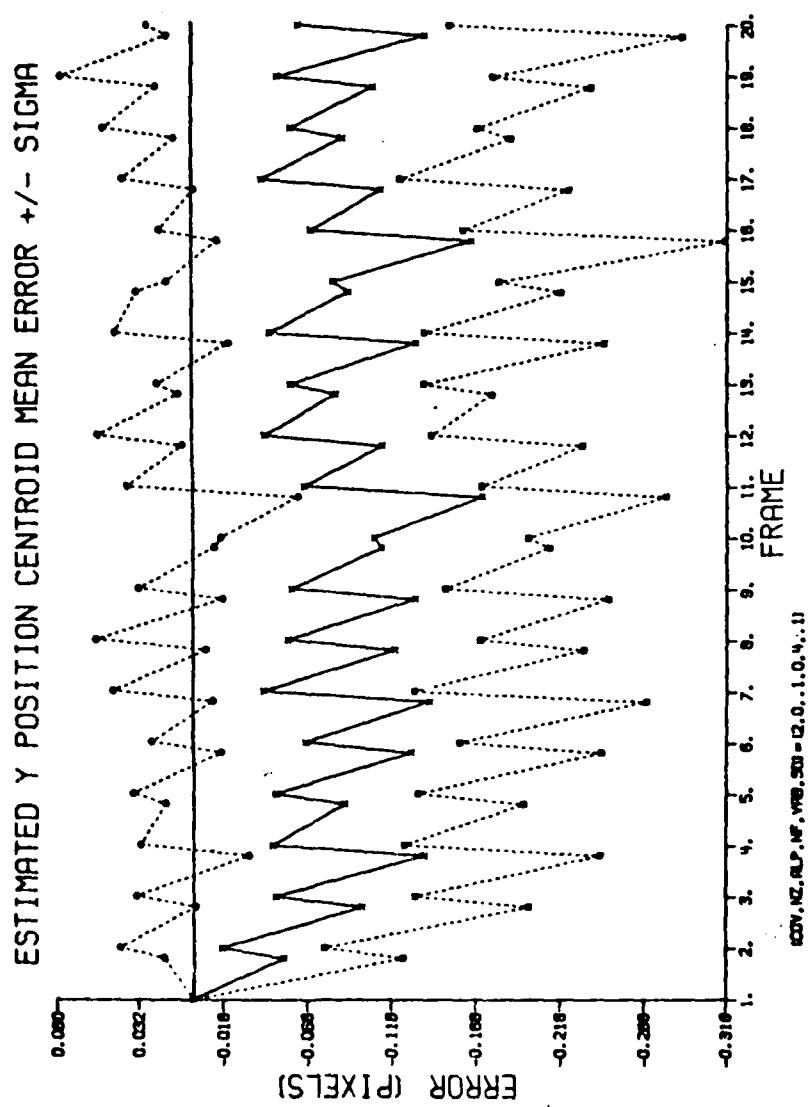


Figure 1-57. Maximum Noise Y Centroid Position Errors

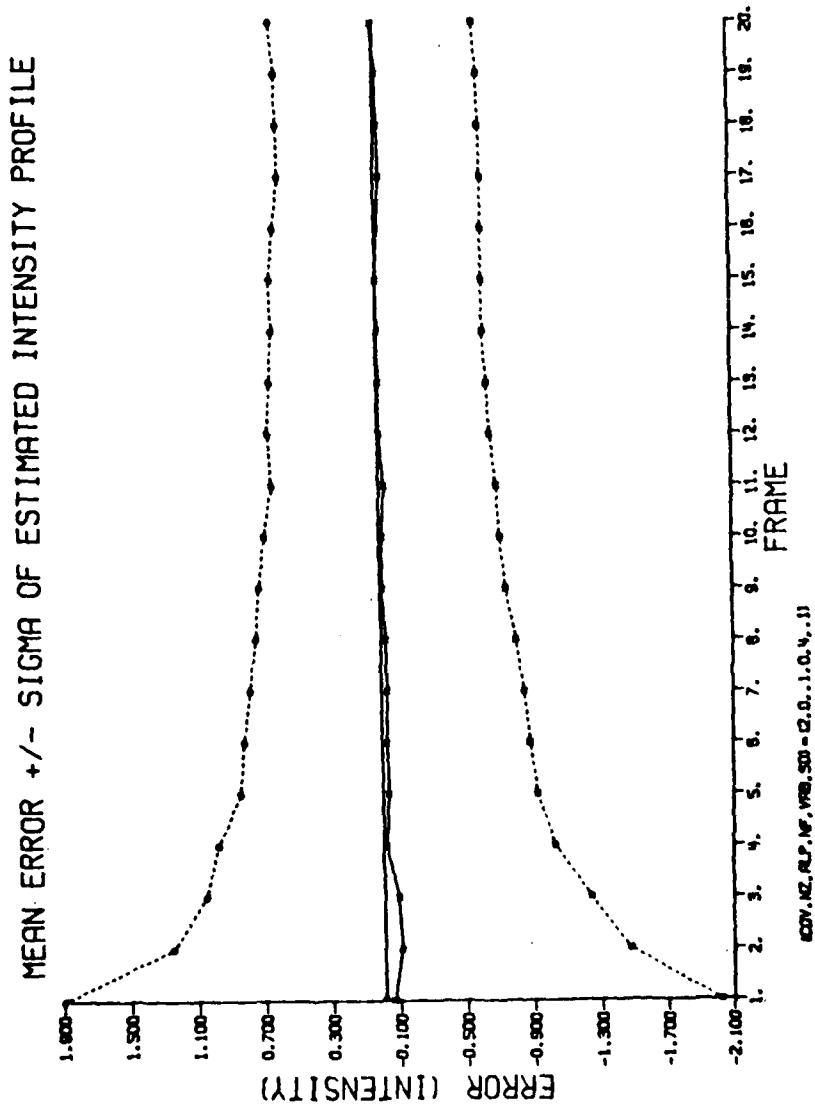


Figure 1-58. Maximum Noise Error of Estimated \hat{h}

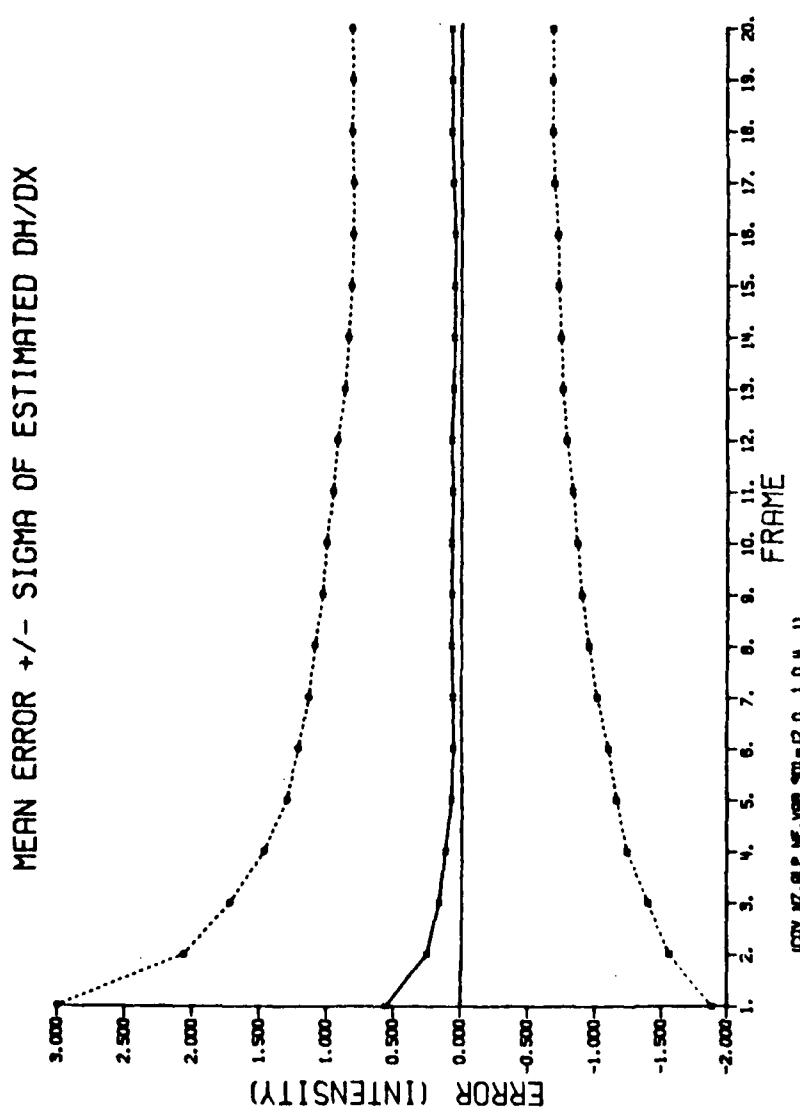


Figure L-59. Maximum Noise Error of Estimated Dh/Dx

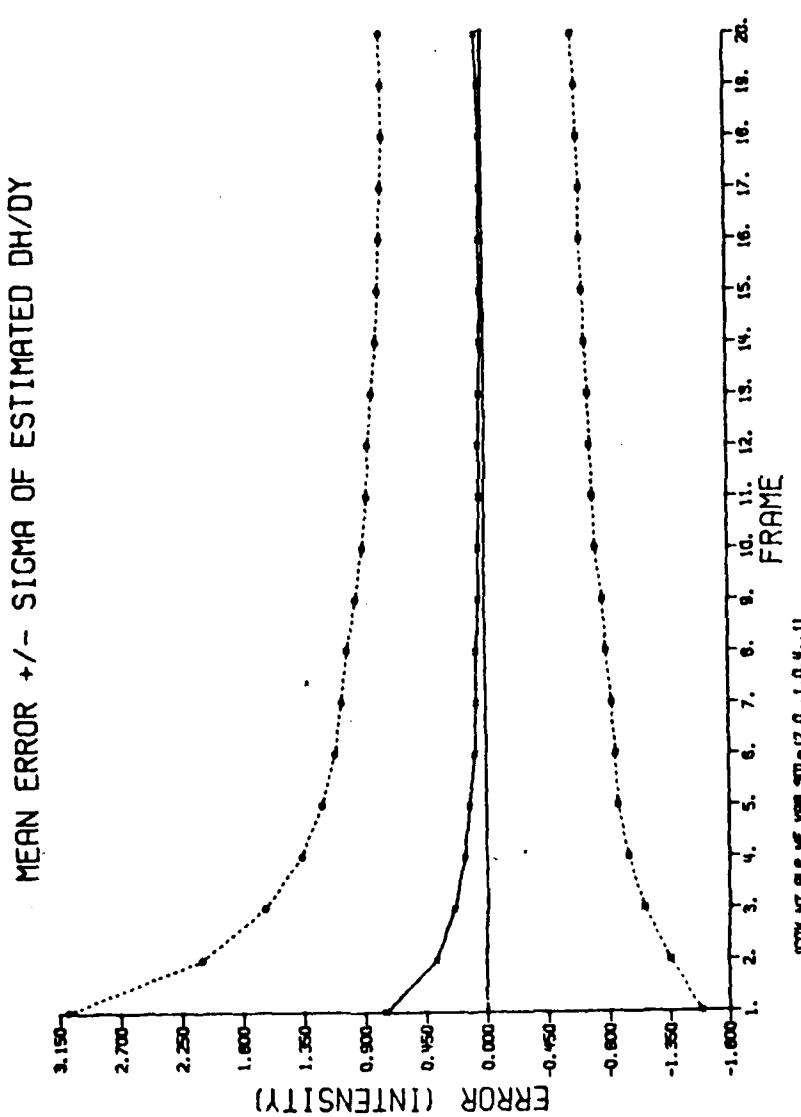


Figure 1-60. Maximum Noise Error of Estimated Dh/Dy

Appendix J

Extended Kalman Filters

This appendix provides an overview of the generic extended Kalman filter algorithm (17). The generic terms of the algorithm are then presented in terms of the tracking example.

The system which contains the states which are to be estimated is assumed to satisfy the nonlinear stochastic differential equation

$$\dot{\underline{x}}(t) = \underline{f}[\underline{x}(t), \underline{u}(t), t] + \underline{G}(t) \underline{w}(t) \quad (J-1)$$

where the initial condition on the state, $\underline{x}(t_0)$, is modelled as a Gaussian random vector with mean $\hat{\underline{x}}_0$ and covariance P_0 . The dynamic driving noise $\underline{w}(t)$ is a zero-mean white Gaussian noise process of strength $Q(t)$.

$$E\{\underline{w}(t) \underline{w}^T(t+\tau)\} = Q(t) \delta(\tau) \quad (J-2)$$

The dynamic driving noise, $\underline{w}(t)$, is assumed to enter in a linear additive fashion in Equation (J-1).

For this application, the generic Equation (J-1) becomes Equation (4-4) where the generic \underline{f} is replaced by a constant \underline{F} . The four filter states, target position in x-y coordinates resulting from dynamics or atmospherics $[x_d \ y_d \ x_a \ y_a]^T$, are multiplied by the constant filter plant matrix \underline{F}_f .

The generic discrete-time measurement equation is

modelled as a nonlinear function of the states and time plus linearly additive noise corruption

$$\underline{z}(t_i) = \underline{h}[\underline{x}(t_i), t_i] + \underline{v}(t_i) \quad (J-3)$$

The measurement corruption noise $\underline{v}(t_i)$ is a white zero-mean Gaussian noise of strength $R(t_i)$ and independent of the dynamic driving noise $\underline{w}(t_i)$ and the initial conditions on the states.

$$E\{\underline{v}(t_i) \underline{v}^T(t_j)\} = \begin{bmatrix} R(t_i) & t_i = t_j \\ 0 & t_i \neq t_j \end{bmatrix} \quad (J-4)$$

In this application the measurement vector, $\underline{z}(t_i)$ is composed of the 64 intensity measurements from the FLIR. The $\underline{h}[\cdot, \cdot]$ function is the expected intensity measurements as a function of the estimated states. The derivation of this intensity function is the subject of Chapter II.

The propagation and measurement update equations for the extended Kalman filter are presented in Chapter IV. These equations are coupled to the state estimate relations via the dependence of $\underline{h}[\cdot, \cdot]$ on the state estimate $\hat{\underline{x}}(t_2)$.

VITA

Steven Keith Rogers was born April 6, 1953 in Jackson, Mississippi. He graduated from E.E. Smith High School in June 1971. In May 1978, he graduated from the University of Colorado with Bachelor degrees in Electrical Engineering and also in Computer Science and entered Air Force Officers Training School that same month. From August 1978 to June 1980, he served as an electrical engineer with the Aeronautical Systems Division at Wright-Patterson Air Force Base, Ohio. In June 1980, Second Lieutenant Steve Rogers was assigned to the Air Force Institute of Technology to pursue a Master's of Science Degree in Electrical Engineering (Electro-Optics). Lt. Rogers is a member of Tau Beta Pi.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GEO/EE/81D-5	2. GOVT ACCESSION NO. <i>AD-A115 581</i>	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) ENHANCED TRACKING OF AIRBORNE TARGETS US- ING FLIR MEASUREMENTS	5. TYPE OF REPORT & PERIOD COVERED MS Thesis	
7. AUTHOR(s) Steven K. Rogers 1Lt, USAF	6. PERFORMING ORG. REPORT NUMBER	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT/EN) Wright-Patterson AFB, Ohio 45433	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS	
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Weapons Laboratory/ARAA Kirtland AFB NM 87117	12. REPORT DATE <i>December 1981</i>	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	13. NUMBER OF PAGES 336	
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited	15. SECURITY CLASS. (of this report) Unclassified	
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) <i>15 APR 1982</i>	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	
18. SUPPLEMENTARY NOTES Approved for public release IAW AFN 100-17 <i>FREDERIC T. LYNCH, Major, USAF</i> <i>S. W. Johnson</i> Director of Public Affairs	Dept. for Research and Professional Development Air Force Institute of Technology (ATC) Wright-Patterson AFB, OH 45433	
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Kalman filter forward looking infrared (FLIR) sensor Atmospheric jitter Correlator tracker 2D Fourier Transforms		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) Considerable work has been accomplished at AFIT in the last three years to improve the tracking capability of the high energy laser weapon. The improvements were achieved via use of an adaptive extended Kalman filter algorithm. In this research, work is initiated on a tracker able to handle "multiple hot spot" targets, in which digital signal processing is employed on the FLIR data to identify the underlying target shape. This identified shape is		

20. ABSTRACT

then used in the measurement model portion of the filter as it estimates target offset from the center of the field-of-view. Two tracking algorithms are developed. The first algorithm uses an extended Kalman filter to process the intensity measurements from a FLIR to produce target position estimates. The second algorithm uses a linear Kalman filter to process the position estimates of an improved correlation algorithm. This algorithm is improved over standard correlators by using thresholding to eliminate poor correlation information, dynamic information from the Kalman filter and it also uses the on-line derived target shape.

UNCLASSIFIED